高等学校计算机应用规划教材

网站设计与Web应用开发技术(第二版)

- 网站策划与设计
- Web服务器的安装与配置
- HTTP协议基础
- HTML开发及编辑工具
- 层叠样式表(CSS)及实例
- JavaScript开发及实例
- 服务器端开发技术基础(.NET及Java等)
- XML、Ajax、开发框架及移动开发技术



吴伟敏　编著

清华大学出版社

高等学校计算机应用规划教材

网站设计与 Web 应用开发技术

(第二版)

吴伟敏　编著

清华大学出版社

北　京

内 容 简 介

本书从 Web 基本概念和网站的规划设计及建设方法入手，着重介绍 HTML、CSS、JavaScript 和服务器端开发技术的基本原理和开发方法，并展望了网站开发领域最新的动向。全书内容在编排上由浅入深，并辅以大量的实例说明。全书共分为 8 章，包括 WWW 简介、网站策划设计与网站运行环境设置、HTTP 协议及其开发与 HTML 语言基础、交互设计及 HTML 高级应用、层叠样式表(CSS)、JavaScript 语言、服务器端开发——动态网页技术基础和 Web 展望。

本书内容丰富，结构清晰，具有很强的实用性，既可作为各类高等院校学习网站设计及 Web 技术的教材，也可作为 Web 开发人员及自学者的参考用书。

本书对应的电子教案、实例源文件和习题答案可以到 http://www.tupwk.com.cn/downpage 网站下载。

图书在版编目(CIP)数据

网站设计与 Web 应用开发技术 / 吴伟敏 编著. —2 版. —北京：清华大学出版社，2015(2018.7 重印)
(高等学校计算机应用规划教材)
ISBN 978-7-302-40023-3

Ⅰ. ①网… Ⅱ. ①吴… Ⅲ. ①网站－设计－高等学校－教材 ②网页制作工具－程序设计－高等学校－教材 Ⅳ. ①TP393.092

中国版本图书馆 CIP 数据核字(2015)第 086749 号

责任编辑：胡辰浩　袁建华
装帧设计：孔祥峰
责任校对：成凤进
责任印制：刘海龙

出版发行：清华大学出版社
网　址：http://www.tup.com.cn，http://www.wqbook.com
地　址：北京清华大学学研大厦 A 座　　邮　编：100084
社 总 机：010-62770175　　邮　购：010-62786544
投稿与读者服务：010-62776969, c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015, zhiliang@tup.tsinghua.edu.cn
课 件 下 载：http://www.tup.com.cn，010-62794504
印 装 者：北京密云胶印厂
经　销：全国新华书店
开　本：185mm×260mm　　印　张：23.5　　字　数：543 千字
版　次：2009 年 1 月第 1 版　　2015 年 5 月第 2 版　　印　次：2018 年 7 月第 3 次印刷
定　价：58.00 元

产品编号：063775-02

前　　言

没有哪一项技术能和今天的 Internet 一样发展迅速，它对人们工作、生活的影响面之广、影响程度之深，使得人们不能不重视它。但是在长期的教学生涯中，笔者注意到虽然很多人希望通过学习掌握技术，但由于没有建立正确的见解和学习的方法，部分人出现了事倍功半的学习结果，乃至于最后不得不放弃。通过观察和分析，希望读者能了解和思考以下几个观点。

1. 对于计算机及其相关技术发展的思考

由于技术的发展会基于越来越高的平台，其发展呈现了不断加速的特征，在这个新思想、新技术以天为单位不断迅速更新的年代，对希望学习网站技术的人提出了更高的要求。因为学习者所面临的是今天所学的技术将不是今后要使用的，而真正需要学的今天还没有出现的现状，对此问题的深入思考一定会帮助读者更好地理解该学什么和该怎么学。如果能透过纷乱的现象看清其中变的与不变的，从更深的层次把握技术的本质，一定能更好地掌握技术的实质并能更好地适应将来的变化。

2. 对于学习方法的思考

网上有大量关于开发的文档，如 HTML、CSS、JavaScript 等，这些知识非常容易获取和查询，但是否获得了这些文档就能成为很好的网站开发者了呢？答案不置可否。虽然在有关文档中所列出的某项功能是确定的，但其用法往往是灵活的，有经验的开发者可以据此而实现多种用途，充分发挥其功用。其实所谓的“经验”是从哪里获取的呢？所谓有经验的人也经历过没有经验的阶段，因而如何快速跨越获取和累积“经验”的鸿沟，是一个值得所有人思考的问题。

基于上面的思考，在本书中将介绍 Web 的发展历史、工作原理、开发框架、网站策划设计、网站安全、HTTP 协议、HTML 语言、层叠样式表(CSS)、CSS 滤镜应用、CSS3 开发、JavaScript 开发、服务器端开发技术基础、XML 技术、Ajax 技术、客户端开发框架以及移动开发等内容。希望这样的内容安排能为大多数希望学习和掌握 Web 技术的读者更好地了解网站及其相关技术的走向和本质有所帮助。对于一个初学者，这本书能引领读者快速入门并迅速成为合格的开发者；对于初级的开发人员，这本书可以答疑解惑提供开发的总体框架和思路，拓展读者的实现手段和方法。

由于本书定位于为今后学习和使用高级的网站开发打下良好的基础，而为了更好地掌握本书所介绍的知识，读者最好能掌握至少一门编程语言。

完整地学习 Web 技术需要具备 3 个层面的知识。本书据此设计了 3 个层次：网站的基本概念及开发基础、Web 技术基础和 Web 高级应用。本书的知识体系结构如图 1 所示，

按照循序渐进的原则，逐步引领读者从基础到各个知识点的学习，为今后的深入学习奠定基础。

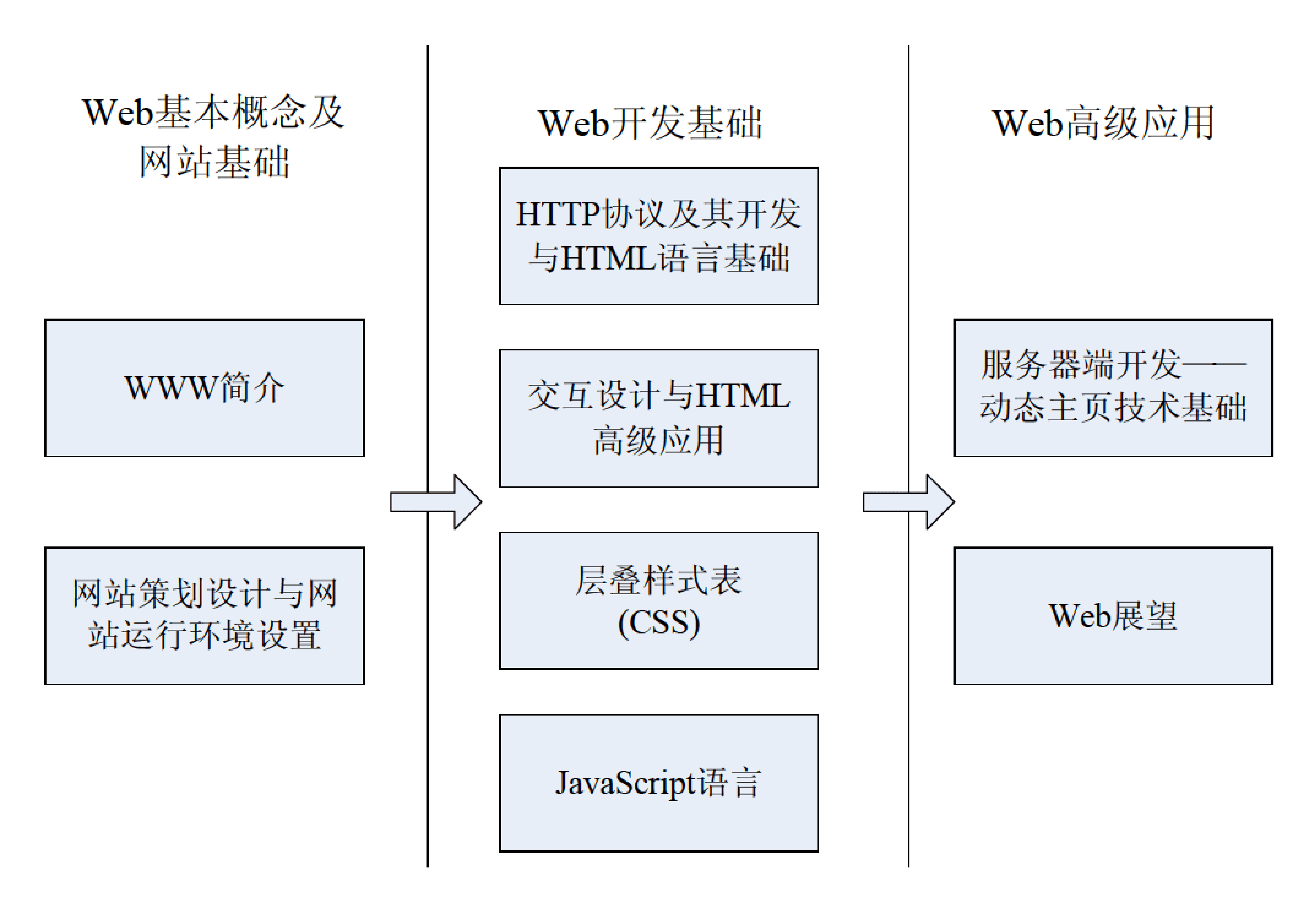


图 1　本书的知识体系结构

概括起来，本书具有以下主要特点。

- 结构清晰、内容详实。在每一章的开始概要说明了本章将介绍的内容，使读者能快速了解本章的要点；介绍每一个知识点时，会辅以实例，先说明此实例的功能、运行的方式，然后给出执行的结果；在各章的最后有对应的小结，总结本章介绍的内容，前后呼应，系统性较强。
- 强调实用性，突出基本原理和方法。为了使读者掌握坚实的基础，学会掌握不断涌现的新技术，本书采用了将网站设计思想与网页制作技术相结合的理念，让读者学会从全局的角度出发考虑和解决当前问题，并通过掌握学习的方法希望读者能解决未来实际工作中遇到的问题。全书按照 Web 开发的方法与顺序，从策划设计入手，循序渐进地介绍了进行 Web 开发的步骤、技巧，并在各章配有精心选择的应用实例，这些实例既有较强的代表性和实用性，又能够综合应用所介绍的知识，使读者能够全面、准确地掌握 Web 开发的全过程，并启发读者达到举一反三的目的。
- 每一章最后附有思考和练习题。这些习题紧扣该章介绍的内容。通过思考和练习能使读者更好地掌握本章介绍的基本概念，提高读者的学习效果和开发技能。

本书共分为 8 章，包括 WWW 简介、网站策划设计与网站运行环境设置、HTTP 协议及其开发与 HTML 语言基础、交互设计与 HTML 高级应用、层叠样式表(CSS)、JavaScript 语言、服务器端开发——动态网页技术基础和 Web 展望。

第 1 章为 WWW 简介，介绍 Internet 与 WWW 的发展历程、网站相关的基本概念及其开发技术以及 Web 的基本框架。第 2 章为网站策划设计与网站运行环境设置，说明在网站

建立之前做好策划工作的必要性，并给出了一些基本原则；为了让网站正确运行，需要在正式开发前做好详细的设计工作；本章还介绍了建立网站开发和运行基本环境的方法。第3章为HTTP协议及其开发与HTML语言基础，介绍了HTTP的基本概念及运行原理、HTML文档的构成和常用元素的基本用法。第4章为交互设计与HTML高级应用，介绍网站交互的设计和实现思路，HTML高级特性和使用方法。第5章为层叠样式表(CSS)，介绍CSS的基本用法、滤镜的使用以及CSS3的基本用法。第6章为JavaScript语言，介绍JavaScript脚本语言的基本概念、基本语法、常用对象和网页特效的制作方法。第7章为服务器端开发——动态主页技术基础，介绍服务器端开发的集中典型方法、动态网页的基本原理以及不同实现技术的特点分析。第8章为Web展望，简单介绍了XML、Ajax、开发框架技术和移动开发的基本特征。

有一定网络和网站基础知识的读者可跳过第1章的学习，具备网站设计、架设和管理经验的读者可跳过第2章的学习。

本书内容安排由浅入深，并注重读者学习和开发能力的培养，通过辅以大量的实例分析和说明，深入、详细地讲解了网站设计与Web应用开发技术，因此本书既可作为各类高等院校学习网站设计及Web技术的教材，也可作为Web开发人员及自学者的参考用书。

本书除封面署名的作者外，南京邮电大学的李啸潇和周骏参与了本书第7章的编写；此外还要感谢负责全书校稿及编辑工作的江苏产业技术研究院的徐欣。

感谢笔者的好友夏兰、徐汝鉴，他们给本书的编写提出了许多指导性的意见；借此还要感谢吴革新、刘迪庐，他们给笔者提出了很多宝贵的建议；另外，为本书编写提供帮助的还有吴殊同、吴晓谦等。正是因为这么多人的大力支持和辛勤汗水，本书才得以出版。

由于本书涉及的内容非常广泛，在深度和广度上很难做到完美，加之笔者水平有限，书中肯定存在错误和不足，请读者批评指正，我们的信箱是 huchenhao@263.net，电话是010-62796045。

本书对应的电子教案、实例源文件和习题答案可以到 http://www.tupwk.com.cn/downpage 网站下载。

作　者

2014年12月

目　　录

第1章　WWW简介

互联网在世界范围内的迅速崛起使得它已经成为一种应用最为广泛的大众传媒，其应用范围正在急剧增长，日益增加的网上购物、各种网络系统和形形色色的网站正在逐渐改变着人们的日常工作、生活、娱乐等行为方式，而其中最为重要的支撑技术就是 Web 技术。

本章旨在引导读者了解 Internet 与 WWW 的发展历程，熟悉 Web 的基本概念及其相关技术；了解开发、运行、调试本书的软硬件环境。本章还将简要介绍各种不同的 Web 开发方法。

本章要点：

- 理解 Internet 与 WWW 的发展历程
- Web 的基本概念
- Web 技术基础及高级技术介绍
- Web 应用开发基础

1.1　Internet 与 WWW

1.1.1　Internet 的发展

诞生于 1946 年的世界上第一台计算机“爱尼亚克”(ENIAC)是一场计算技术的革命，数字信息时代也由此拉开了序幕。在之后的若干年中，计算机的处理能力基本按照每 18 个月就翻一番的规律发展，由于这个定律首先是由美国英特尔公司的戈登 • 摩尔提出并应用的，因此这个定律被称为“摩尔定律”。

早期的计算机是独立的，之后为了能在计算机之间方便地进行通信和共享资源，诞生了网络，由此宣告了网络时代的到来。Internet 最早来源于美国国防部高级研究计划局 DARPA(Defense Advanced Research Projects Agency)的前身 ARPA 建立的 ARPAnet，它源于当时美国国防部为了保证美国国防力量在受到第一次核打击后仍能具有生存和反击的能力而设计的分散指挥系统。该网于 1969 年投入使用，最初由加州大学、犹他大学和斯坦福研究院的 4 台计算机以分组交换的原理构成。从 20 世纪 60 年代开始，ARPA 就开始向美国国内大学的计算机系和一些私人有限公司提供经费，以促进基于分组交换技术的计算机网络的研究。1968 年，ARPA 为 ARPAnet 网络项目立项，这个项目基于这样一种主导思想：网络必须能够经受住故障的考验并维持正常工作，一旦发生战争，当网络的某一部分因遭受攻击而失去工作能力时，网络的其他部分应当能够维持正常通信。最初，ARPAnet

主要用于军事研究目的，它具有五大特点：

- 支持资源共享；
- 采用分布式控制技术；
- 采用分组交换技术；
- 使用通信控制处理机；
- 采用分层的网络通信协议。

1972 年，ARPAnet 在首届计算机后台通信国际会议上首次与公众见面，并验证了分组交换技术的可行性，由此，ARPAnet 成为现代计算机网络诞生的标志。

ARPAnet 在技术上的另一个重大贡献是 TCP/IP 协议簇的开发和使用。1980 年，ARPA 投资把 TCP/IP 加进 UNIX(BSD4.1 版本)的内核中，在 BSD4.2 版本以后，TCP/IP 协议即成为 UNIX 操作系统的标准通信模块。1982 年，Internet 由 ARPAnet、MILNET 等几个计算机网络合并而成。作为 Internet 的早期骨干网，ARPAnet 奠定了 Internet 存在和发展的基础，较好地解决了异构环境下网络互联的一系列理论和技术问题。

1983 年，ARPAnet 分裂为两部分：ARPAnet 和纯军事用的 MILNET。同年 1 月，ARPA 把 TCP/IP 协议作为 ARPAnet 的标准协议，其后，人们称呼这个以 ARPAnet 为主干网的网际互联网为 Internet，TCP/IP 协议簇便在 Internet 中进行研究、试验，并改进成为使用方便、效率极好的协议簇。

与此同时，局域网和其他广域网的产生和蓬勃发展对 Internet 的进一步发展起到了重要的作用。其中，最引人注目的就是美国国家科学基金会 NSF(National Science Foundation)建立的美国国家科学基金网 NSFnet。1986 年，NSF 建立起了六大超级计算机中心，为了使全国的科学家、工程师能够共享这些超级计算机设施，NSF 建立了自己的基于 TCP/IP 协议簇的计算机网络 NSFnet。NSF 在全国建立了按地区划分的计算机广域网，并将这些地区网络和超级计算中心相连，最后将各超级计算中心互联起来。地区网的构成一般是由一批在地理上局限于某一地域，在管理上隶属于某一机构或在经济上有共同利益的用户的计算机互联而成，连接各地区网上主通信节点计算机的高速数据专线构成了 NSFnet 的主干网，这样，当一个用户的计算机与某一地区相连以后，它除了可以使用任一超级计算中心的设施，可以同网上任一用户通信，还可以获得网络提供的大量信息和数据。这一成功使得 NSFnet 于 1990 年 6 月彻底取代了 ARPAnet 而成为 Internet 的主干网。

NSFnet 对 Internet 的最大贡献是使 Internet 向全社会开放，而不再像以前那样仅仅为计算机研究人员、政府职员和政府承包商所使用。然而，随着网上通信量的迅猛增长，NSF 不得不采用更新的网络技术来适应发展的需要。1990 年 9 月，由 Merit、IBM 和 MCI 公司联合建立了一个非营利性的组织——先进网络和科学公司 ANS(Advanced Network Science，Inc)。ANS 的目的是建立一个全美范围的 T3 级主干网，它能以 45Mb/s 的速率传送数据，相当于每秒传送 1400 页文本信息。到 1991 年底，NSFnet 的全部主干网都已同 ANS 提供的 T3 级主干网相通。

1969 年 12 月，当 ARPAnet 最初建成时只有 4 个节点，到 1972 年 3 月也仅增加到 23 个节点，直到 1977 年 3 月总共也只有 111 个节点。但是近十年来，随着社会科技、文化和

经济的发展，特别是计算机网络技术和通信技术的大发展，以及人类社会从工业社会向信息社会过渡的趋势越来越明显，人们对信息的认识，对开发和使用信息资源的重视越来越强烈，这些都强烈刺激了 ARPAnet 和以后发展成的 NSFnet 的发展，使连入这两个网络的主机和用户数目急剧增加。1988 年，由 NSFnet 连接的计算机数就猛增到 56000 台，此后每年更以 2~3 倍的惊人速度向前发展；1994 年，Internet 上的主机数目达到了 320 万台，连接了世界上的 35000 个计算机网络；2000 年，全球已有超过一亿名用户，而这个数字此后以每年 15%~20%的速度递增。中国互联网络信息中心的数据显示，截至 2014 年 6 月，中国的互联网用户数达 6.86 亿，中国是全球最大的互联网市场，而且未来这个数量还将以更快的速度增加。Internet 发展过程中的重要阶段如表 1-1 所示。

表 1-1　Internet 发展过程中的重要阶段

	1969 年	1982 年	1986 年	80 年代后期
网络名称	ARPANET(美国国防部高级研究计划署网)	ARPANET 与 MILNET 合并形成 Internet 雏形	NSFNET(国家科学基金网)取代 ARPANET 成为 Internet 基础	Internet 形成并迅速发展

在 Internet 蓬勃发展的同时，其本身随着用户需求的转移也不断发生着产品结构上的变化，现已成为全球重要的信息传播工具。我国于 1994 年 5 月正式接通 Internet，发展至今已将近 20 年的时间。据 2014 年中国互联网络信息中心(CNNIC)在北京发布的《第 31 次中国互联网络发展状况统计报告》，截至 2013 年 12 月底，我国网民规模达 6.18 亿，全年共计新增网民 5358 万人，互联网普及率为 45.8%，较 2012 年底提升 3.7 个百分点，如图 1-1 所示；中国域名总数为 1844 万个；家中使用计算机接入互联网网民比例 90.3%；农村网民规模达 1.77 亿，同比增加 2101 万；网民平均每周上网时长为 25 小时；网上支付用户规模达 2.6 亿，使用率提升至 42.1%；即时通信用户规模达 5.32 亿，同比增长 6440 万人。

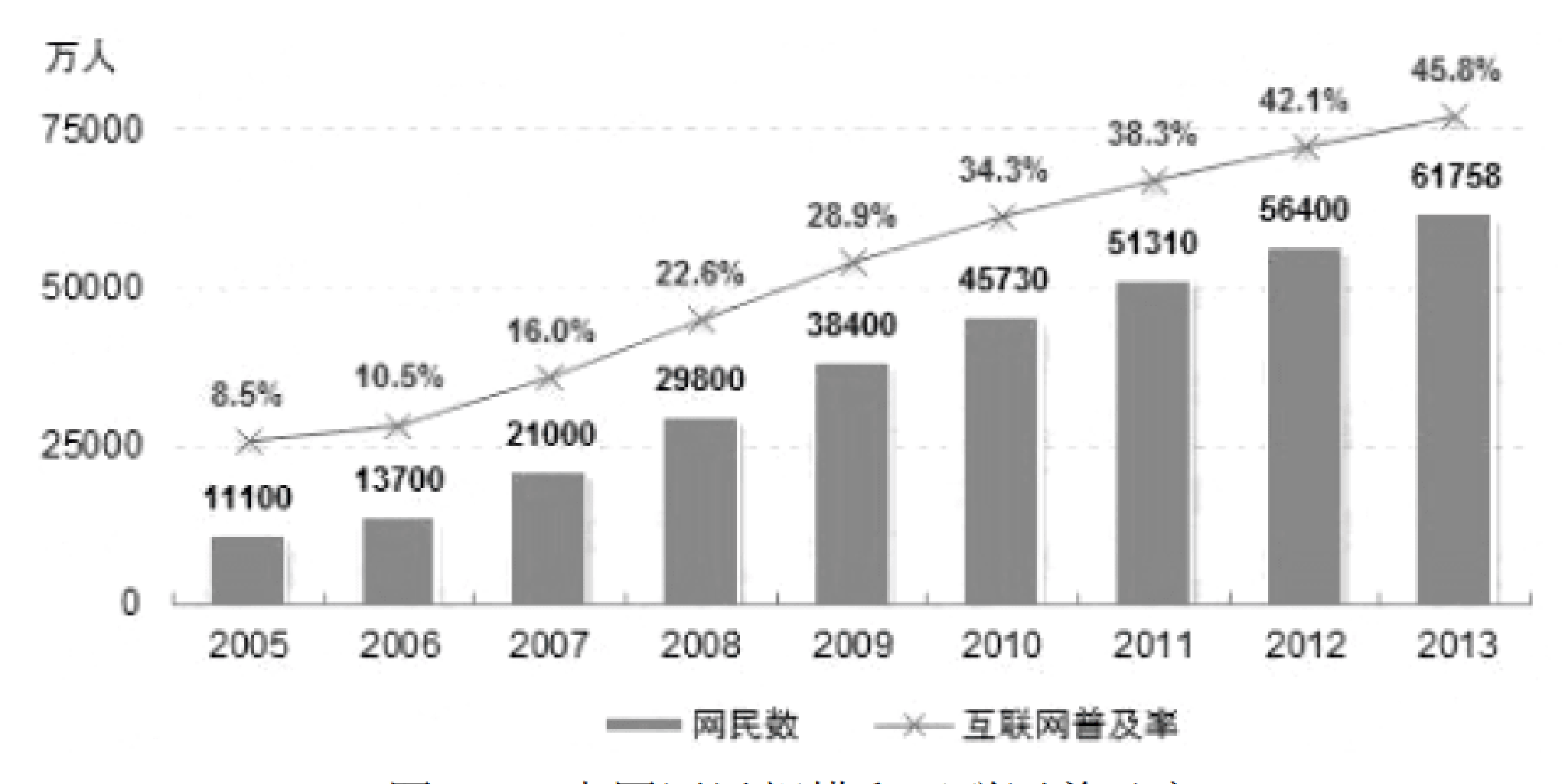


图 1-1　中国网民规模和互联网普及率

另外一个值得注意的现象是：手机网民保持快速增长，69.7%的网民通过台式电脑上网，相比 2012 年底下降了近 1 个百分点；通过笔记本电脑上网的网民比例与 2012 年底相比略有降低，为 44.1%；而手机上网的比例保持较快增速，从 74.5%上升至 81.0%，如图 1-2 所示。

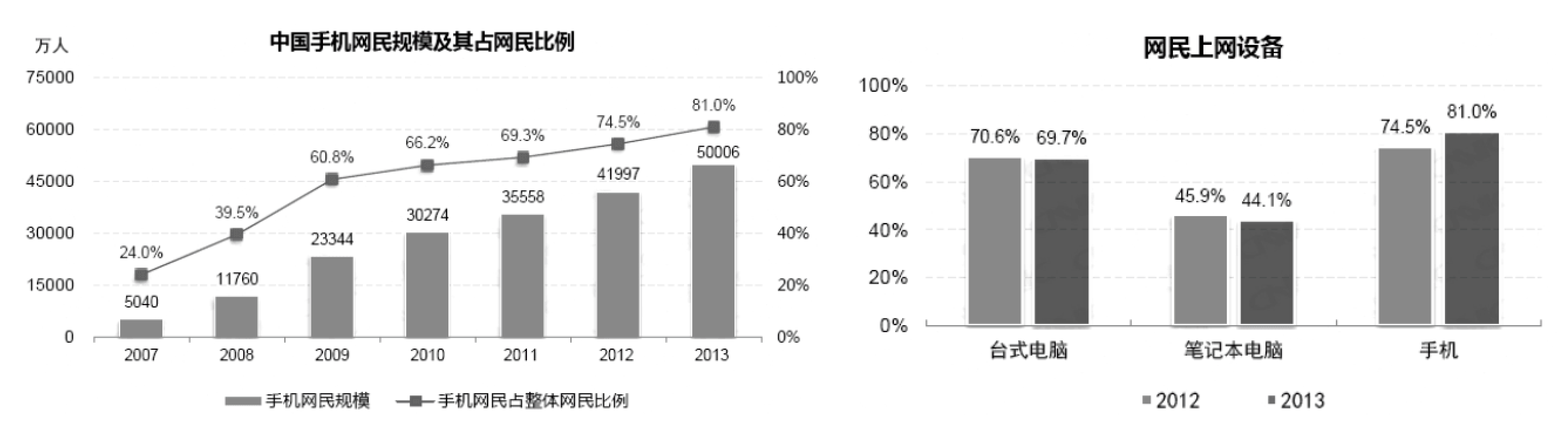


图 1-2　网民上网所使用的设备

在 Internet 上，按从事的业务分类包括了广告公司、航空公司、农业生产公司、艺术、导航设备、书店、化工、通信、计算机、咨询、娱乐、财贸、各类商店、旅馆等 100 多类，覆盖了社会生活的方方面面，构成了一个信息社会的缩影。由于越来越多计算机的加入，Internet 上的资源变得越来越丰富。到今天，Internet 已超出一般计算机网络的概念，它不仅是传输信息的媒体，而且成为一个全球规模的信息服务系统。它是人类有史以来第一个真正的世界性的“图书馆”，又是一个全球范围的论坛。

1.1.2　Internet 技术基础

1. TCP/IP

1972 年出现了网际互联的核心技术 TCP/IP 协议，该协议包括近 100 个协议，而其中最主要的是 TCP 协议和 IP 协议，其中 TCP(Transmission Control Protocol)是传送控制协议，它的作用是保证信息在网络间可靠地传送，保证接收到的信息在传送途中不被损坏；而 IP(Internet Protocol)是网际网协议，保证信息从一个地方传送到另一个地方，不管中间要经过多少节点和不同的网络。TCP/IP 模型的网络协议如图 1-3 所示。

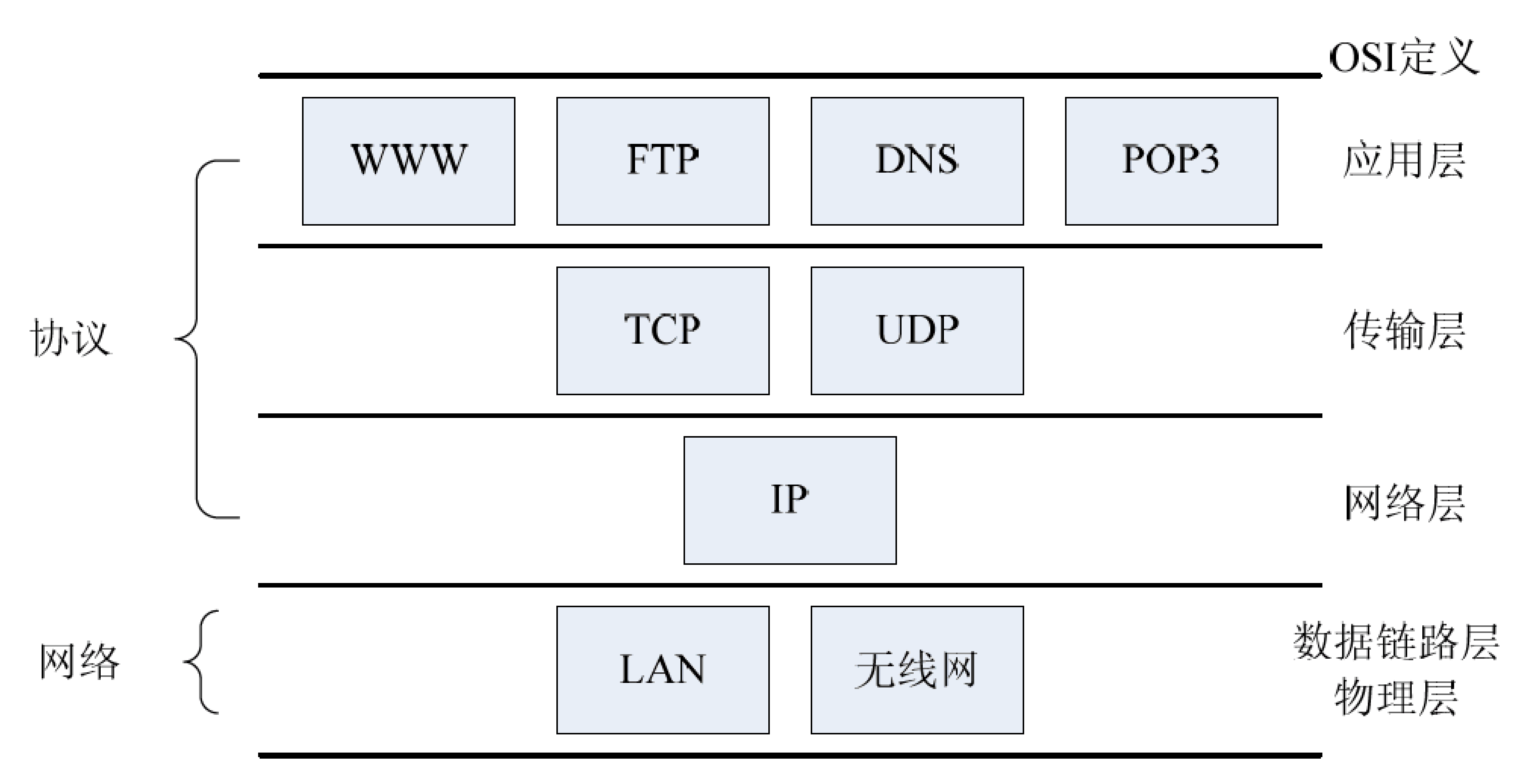


图 1-3　TCP/IP 模型的网络协议

IP 是 TCP/IP 体系结构中非常重要的协议，它是为计算机网络相互连接进行通信而设计的协议，IP 协议是基于分组交换技术的，它包含如下规则。

- 目前，Internet 上每台计算机都有一个由 4 个数字组成的 Internet 地址，每个不超过 256，地址数码用点分开，如 202.96.101.201。

- 一个信息被划分成若干个分组。
- 每个分组被填入一个 IP 信封。
- IP 信封外包含一个发送地址和一个收信地址，再加一个顺序号。

在 Internet 上每台主机都有专门的地址，称为 IP 地址，只有有了地址，信息才可以传送，这正如日常生活中发送纸质邮件需要地址一样。在 Internet 中，IP 地址用 4 个数字组成，每个数字不大于 256，数字间用.分开；在计算机中 IP 地址是从左到右表示的，最左边部分识别网络中的最大部分，IP 是由管理 IP 地址的专门机构分配的。

在互联网中，IP 协议是能使连接到网上的所有计算机网络实现相互通信的一套规则，规定了计算机在互联网上进行通信时应当遵守的规则。任何厂家生产的计算机系统，只要遵守 IP 协议就可以与互联网互连互通。

通俗的讲，为了访问互联网中的计算机，必须有一种寻址方法来定位，IP 地址就是互联网上的主机和路由器的一种标识方法。每个互联网上的主机和路由器都有一个 IP 地址，它包括网络号和主机号。这一编码组合是唯一的，没有两台有同一 IP 地址的计算机。

2. IPv6

IPv6(Internet Protocol Version 6)是 IETF(Internet Engineering Task Force，互联网工程任务组)设计的用于替代现行版本 IP 协议(IPv4)的下一代 IP 协议。

当前所使用的第二代互联网 IPv4 技术，其核心技术属于美国。它的最大问题是网络地址资源有限，从理论上讲，IPv4 可以实现为 1600 万个网络和共计 40 亿台主机编址。但采用 A、B、C 三类编址方式后，可用的网络地址和主机地址的数目大打折扣。实际上，IPv4 的地址已于 2011 年 2 月 3 日分配完毕，其中北美占有 3/4，约 30 亿个，而人口最多的亚洲只有不到 4 亿个。

一方面是地址资源数量的限制，另一方面是随着电子技术及网络技术的发展，物联网时代的到来将可能使人们身边的每一样东西都连入互联网。在这种需求的推动下，IPv6 应运而生。单从数量级上来说，IPv6 所拥有的地址容量是 IPv4 的约 8×10^28 倍，达到 2^128(算上地址为全零的和全部 255)个。这不但解决了网络地址资源数量的问题，同时也为物联网的推进在 IP 地址数量限制的问题上扫清了障碍。

由于 Internet 的规模以及网络中数量庞大的 IPv4 用户和设备，IPv4 到 IPv6 的过渡不可能一次性实现。而且，许多企业和用户的日常工作越来越依赖于 Internet，它们无法容忍在协议过渡过程中出现的问题。所以 IPv4 到 IPv6 的过渡必须是一个循序渐进的过程，在体验 IPv6 带来的好处的同时仍能兼容网络中原先使用 IPv4 的设备。实际上，IPv6 在设计的过程中就已经考虑到了 IPv4 到 IPv6 的过渡问题，来简化过渡过程。

3. 域名系统(DNS)

如果使用 Internet 就必须使用 IP 地址，那么这个经历将是非常痛苦的。值得庆幸的是，作为一个 World Wide Web 的用户，实际上并不需要对 IP 地址有很深的了解，也不需要记住很多枯燥的 IP 地址，这应归因于一种 Internet 上的计算机的命名方案，我们称之为域名

系统(Domain Name System，简称为 DNS)。它可以将形如 www.njupt.edu.cn 的域名与其所对应的 IP 地址进行对应和转换。因此，我们就可以使用域名来取代 IP 地址了。在语法上，每台计算机的域名由一系列字母和数字构成的段组成。例如，某个服务器的域名为 www.njupt.edu.cn，其中，cn 代表中国，edu 代表教育部门，njupt 代表南京邮电大学，www 代表 WWW 服务。

DNS 是一个分布式的数据库，利用 DNS 能进行域名的解析，一般是存放于 DNS 服务器上，为了定义 Internet 上的主机而提供的一个层次性的命名系统，如图 1-4 所示。

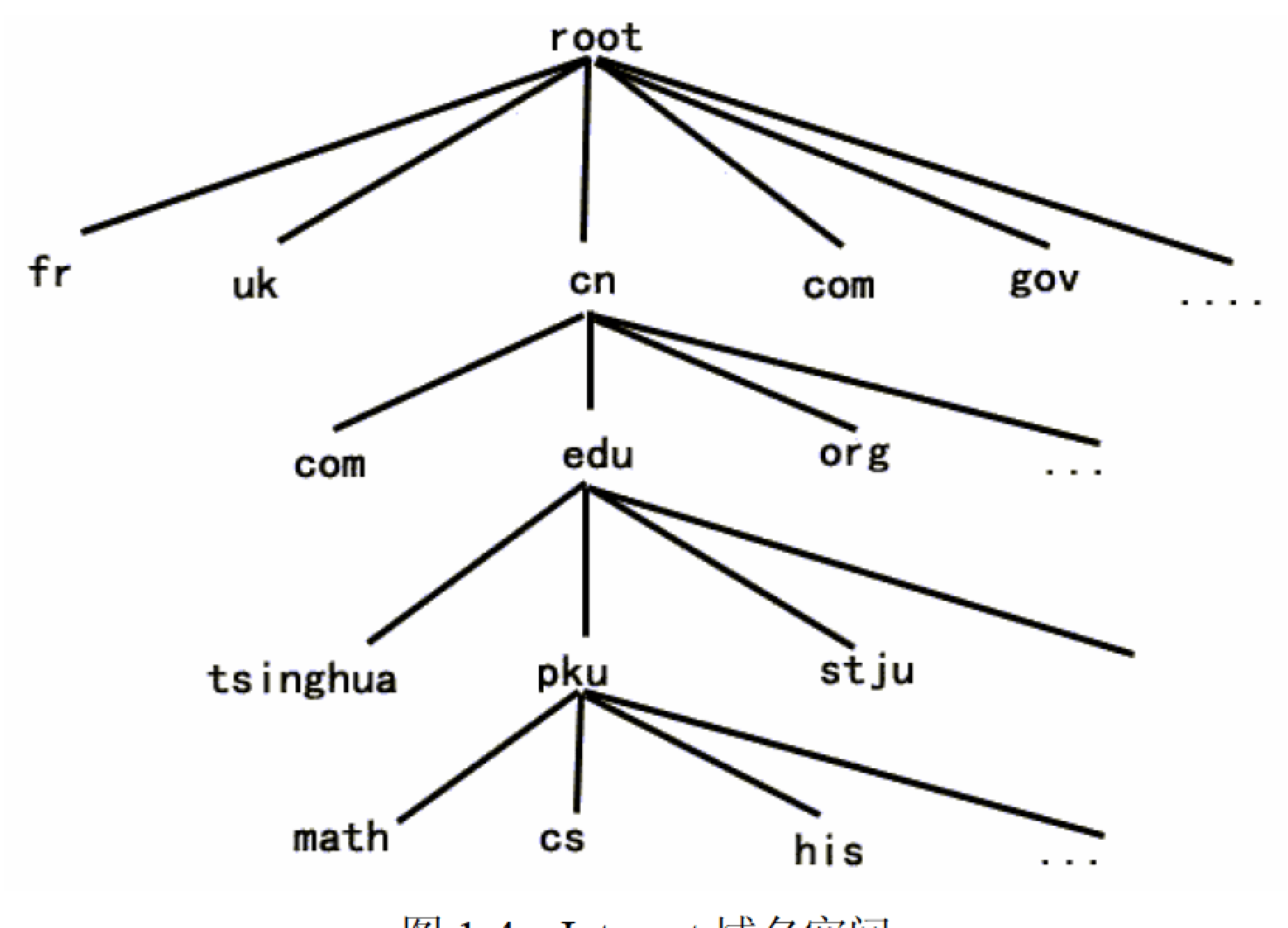


图 1-4　Internet 域名空间

域名的解析过程是这样的：

- DNS 客户向本地的 DNS 服务器发出查询请求。
- 如果该 DNS 本身具有客户想要查询的数据，则直接返回给客户，如果没有，则该服务器和其他命名服务器联系，从其他服务器上获取信息，然后返回给用户。

各种域名扩展名是有意义的，域名系统(Domain Name System，DNS)常见的扩展名及其含义如表 1-2 所示。

表 1-2　常见域名系统的含义

域名扩展名	含　　义
edu	教育及学术单位
com	公司或商业组织
gov	政府单位
mil	军事单位
org	基金会等非官方单位
net	网络管理服务机构
int	国际性组织
apra	APRANET Internet 的起源
国别名(国家/及地区代码)	依 ISO 标准定义，例如，cn 代表中国

1.1.3 Internet 提供的服务

Internet 的飞速发展和广泛应用得益于它所提供的多种服务，这些服务为人们的信息交流带来了极大的便利，下面介绍 Internet 所提供的几种主要服务。

1. WWW 服务

WWW(World Wide Web，环球信息网)是一个基于超文本方式的信息查询方式。WWW 是由欧洲粒子物理研究中心(CERN)研制的。通过超文本方式将 Internet 上不同地址的信息有机地组织在一起，WWW 提供了一个友好的界面，大大方便了人们的信息浏览，而且 WWW 方式仍然可以提供传统的 Internet 服务，如 Telnet、FTP、Gopher、News、E-Mail 等。

很多相关的服务实际上就是借助 WWW 来实现的，包括搜索引擎、网络新闻、博客、网络视频、网络游戏、微博、社交网站、网络购物、网上银行、论坛、Web 邮件、网上支付、网上炒股等。

2. 文件传输服务(FTP)

FTP 服务解决了远程传输文件的问题，无论两台计算机相距多远，只要它们都连入 Internet 并且都支持 FTP 协议，则这两台计算机之间就可以进行文件的传送。FTP 实质上是一种实时的联机服务，用户首先要登录到目标服务器上，之后用户可以在服务器目录下寻找所需文件，FTP 几乎可以传送任何类型的文件，如文本文件、二进制文件、图像文件、声音文件等。一般的 FTP 服务器都支持匿名(Anonymous)登录，用户在登录到这些服务器时无须事先注册用户名和口令，只要以 anonymou 为用户名和合法的 E-mail 地址作为口令就可以访问该 FTP 服务器了。

3. 电子邮件服务(E-mail)

电子邮件(E-mail)是Internet上使用最广泛和最受欢迎的服务，它是网络用户之间进行快速、简便、可靠且低成本联络的现代通信手段。电子邮件使网络用户能够发送和接收文字、图像和语音等多种形式的信息。使用电子邮件的前提是拥有自己的电子信箱，即E-mail地址，实际上是在邮件服务器上建立一个用于存储邮件的磁盘空间。电子邮件地址的典型格式为username@mailserver.com，其中mailserver.com部分代表邮件服务器的域名，username代表用户名，符号@读作“at”，意为“在”。例如某E-mail地址为master@njupt.edu.cn，其含义表示为在计算机njupt.edu.cn上用户名为master的电子邮件地址。利用电子邮件可以获得其他各种服务(如FTP、Gopher、Archie、WAIS等)。当用户希望从这些信息中心查询资料时，只需要向其指定的电子信箱发一封含有一系列信息查询命令的电子邮件，该邮件服务器程序将自动读取、分析该邮件中的命令，若无错误则将检索结果通过邮件方式发给用户。

4. 远程登录服务(Telnet)

远程登录是 Internet 提供的最基本的信息服务之一，Internet 用户的远程登录是在网络通信 Telnet 的支持下使自己的计算机暂时成为远程计算机仿真终端的过程，要在远程计算机上登录，首先应给出远程计算机的域名或 IP 地址。另外，事先应该成为该远程计算机系

统的合法用户并拥有相应的账号和口令。目前国内 Telnet 最广泛的应用就是 BBS(电子公告牌)，通过 BBS 用户可以进行各种信息交流、讨论。

5. 视音频业务

基于数字视频通信会议电视已经发展了多年，在视频点播、远程教育、视频监控、视频会议、Internet 直播方面有了广泛的应用。由于 Internet 的无连接数据包转发机制主要为突发性的数据传输设计，不适用于对连续媒体流的传输，因此为了在 Internet 上有效、高质量地传输视频流，需要多种技术的支持，主要包括视频的压缩、编码技术，应用层质量控制技术，连续媒体分布服务技术，媒体同步技术和数字版权管理技术等。

基于 Internet 的语音传输是利用基于 IP 数据网进行的话音传输。话音(模拟信号)首先由数字信号处理器(DSP)将其转换为数字信号，然后，数字信号被压缩成更便于网络传输的数据包，之后，通过 Internet 将数据包传送到目的地，在目的地以相反的过程解压缩、解包、数/模转换，送达对方话筒。由于 Internet 中采用“存储——转发”的方式传递数据包，并不独占电路，并且对语音信号进行了大比例的压缩处理，因此，IP 电话占用带宽仅为 8~10kb/s，还不到模拟电话所需带宽的 1/8，再加上 Internet 上数据传输的计费方式与距离的远近无关，就大大节省了长途通信费用。

6. 电子商务

电子商务是指利用计算机网络进行的商务活动，它将顾客、销售商、供货商和雇员联系在一起，实现商务活动的电子化、网络化、自动化。在互联网开放的网络环境下，买卖双方在任何可连接网络的地点间进行各种商务活动，实现两个或多个交易者间的生产资料交换及所衍生出来的交易过程、金融活动和相关的综合服务活动的一种商业运营模式。

7. 对等网服务(P2P)

P2P 是英文 Peer-to-Peer(对等)的简称，有时也被称为“点对点”。“对等”技术，是一种网络新技术，依赖网络中参与者的计算能力和带宽，而不是依赖于较少的几台服务器上。目前人们认为其在加强网络上人的交流、文件交换、分布计算等方面大有前途。

简单的说，P2P 直接将人们联系起来，让人们通过互联网直接交互。P2P 使得网络上的沟通变得容易、更直接共享和交互，真正地消除中间商。人们可以直接连接到其他用户的计算机、交换文件，而不是像过去那样连接到服务器去浏览、下载与交换。P2P 另一个重要特点是改变互联网现在的以大网站为中心的状态，重返“非中心化”，并把权力交还给用户。P2P 看起来似乎很新，但是正如 B2C、B2B 是将现实世界中很平常的东西移植到互联网上一样，P2P 并不是什么新东西。在现实生活中我们每天都按照 P2P 模式面对面地或者通过电话交流和沟通。

即使从网络看，P2P 也不是新概念，P2P 是互联网整体架构的基础。互联网最基本的协议 TCP/IP 并没有客户机和服务器的概念，所有的设备都是通信的平等的一端。在十年之前，所有的互联网上的系统都同时具有服务器和客户机的功能。当然，后来发展的那些架构在 TCP/IP 之上的软件的确采用了客户机/服务器的结构：浏览器和 Web 服务器、邮件

客户端和邮件服务器。但是，对于服务器来说，它们之间仍然是对等联网的。以 E-mail 为例，互联网上并没有一个巨大的、唯一的邮件服务器来处理所有的 E-mail，而是对等联网的邮件服务器相互协作把 E-mail 传送到相应的服务器上去。另外用户之间 E-mail 则一直对等地联络渠道。

事实上，网络上现有的许多服务都可以归入 P2P 的行列。即时讯息系统，如 ICQ、AOL Instant Messenger、Yahoo Pager、微软的 MSN Messenger 以及国内的 QQ 等；下载工具，如 BitTorrent、BitSpirit、eMule(电驴)、百度下吧、PP 点点通、卡盟、迅雷等；视频传输工具，如 PPFilm、PPVod、QQlive、各种 P2P 网络电视软件等，都是流行的 P2P 应用。它们允许用户互相沟通和交换信息、交换文件，甚至于实现远程协助等复杂应用。

P2P 网络的一个重要的目标就是让所有的客户端都能提供资源，包括带宽、存储空间和计算能力。因此，当有节点加入且对系统请求增多，整个系统的容量也增大。这是具有一组固定服务器的 Client-Server 结构不能实现的，因为在上述这种结构中，客户端的增加意味着所有用户更慢的数据传输。 P2P 网络的分布特性通过在多节点上复制数据，也增加了防故障的健壮性，并且在纯 P2P 网络中，节点不需要依靠一个中心索引服务器来发现数据。在后一种情况下，系统也不会出现单点崩溃。

在具有上述优点的同时，P2P 技术也有流量大、占用网络大量带宽的缺点，但以下的技术可以使这个问题在一定程度上得到缓解。

- P4P(Proactive network Provider Participation for P2P)技术，这项技术其实是 P2P 技术的升级版，目的是为了加强 ISP 与客户端程序的通信，降低骨干网的数据传输压力，并提高文件传输的性能。P4P 与 P2P 最大的不同在于它可以有针对性地选择传输节点，而不是像 P2P 那样，随机选择。这样就可以把 P2P 节点的传输区域控制在某个范围，最大限度地解决大型节点和网络出口负载，从而缓解骨干网的拥堵。
- PCDN 技术。这项技术在 CDN 节点的边缘构建了基于用户的 P2P 自治域，通过集中的分布式架构将 P2P 的流量严格限制在同一边缘节点的区域内。这项技术的原理与 P4P 技术非常相似，即通过控制 P2P 流量传输的范围而降低其对骨干网的挑战。
- P2P 服务器模式。即把服务器而不是 PC 当成 CDN 网络的节点，达到 CDN 网络优化和加速的目的。服务器之间实现 P2P 连接，这样就不用再到中心节点的存储上寻找内容，从而提高网络传输的效率。

1.2　WWW 概述

1.2.1　WWW 的起源

Web 源于欧洲核能研究中心(CERN)的 TIM BERNERS-LEE 于 1989 年提出的链接文档构想，由日内瓦粒子物理实验室发明。后来它在 TCP/IP、MIME、Hypertext 等技术之上发展起来，并开发了 HTTP(Hypertext Transfer Protocal)、HTML(Hypertext Markup Language)、

URL(Uniform Resource Location)等多项新技术。

那么，什么是 Web 呢？Web 是 World Wide Web 的简称，Web 的本意是蜘蛛网和网的意思，在网页设计中被称为网页，中译为“万维网”，现广泛译作网络、互联网等技术领域。实际上 Web 是运行在 Internet 之上的所有 HTTP 服务器软件和它们所管理的对象的集合。这个对象包括了 Web Page/Web 文档和程序，由于 Web 技术涉及的面很广，为了能有一个比较清楚的认识，在此首先对 Web 的历史进行简单介绍。

Web 现在变得越来越复杂，但刚开始时一切都那么简单。在美国，最初为了连接很少的几个顶尖研究机构，设计了最早的“Internet”，以便共同开展科学研究。不论是图书馆员、原子能物理学家，还是计算机科学家，都必须学习一个相当复杂的系统，1962 年，麻省理工大学(MIT)的 J.C.R. Licklider 最早提出他的“Galactic Network”(超大网络)思想——设想了全球计算机互联的一系列概念，其中的资源和信息能够在任何站点上被处理。这个简单的设想经过发展和多年的努力，最终形成了现在的 Web。

最初，研究人员认为传输控制协议(Transmission Control Protocol，TCP)只适用于大型系统，因为 TCP 就是为大型系统设计的。不过，麻省理工大学 David Clark 的研究小组却发现，这个协议也可以在工作站之间实现大面积的互联。Clark 的这项研究为 Web 的发展解决了底层网络通信的问题，为将来 Web 的广泛使用奠定了基础。

正如上文所述，随着主机数量从为数不多的几个发展到成千上万，去记忆数量众多且毫无意义的数字地址编号就显得极其麻烦，人们开始设想为主机指定不同的名字来解决上面的问题。这造就了域名系统(Domain Name System，简称 DNS)。另外，ARPANET 决定从使用网络控制协议(Network Control Protocol，简称 NCP)变为使用 TCP/IP(Transmission Control Protocol/Internet Protocol，中译为传输控制协议/ Internet 协议)，而 TCP/IP 是军方使用的标准协议。到了 20 世纪 80 年代中期，Internet 已经建成为一个连接不同研究人员的平台，而且其他网络也开始出现：如美国国家航空和宇宙航行局(National Aeronautics and Space Administration)创建了 SPAN、美国能源部(U.S. Department of Energy)建立了 MFENet 等。

1980 年欧洲粒子物理研究所(European Organization for Nuclear Research，简称 CERN)的 Tim Berners-Lee 负责了 Enquire(Enquire Within Upon Everything)项目。1989 年， Tim Berners-Lee 提出了一个很有意思的概念：他认为，与其简单地引用其他人的工作，为什么不干脆采用链接呢?读一篇文章时，科学家可以直接打开所引用的文章。

超文本(HyperText)当时相当流行，并利用了他先前在文档和文本处理方面的研究成果。Berners-Lee 发明了标准通用标记语言(Standard Generalized Markup Language，SGML)的一个子集，称为超文本标记语言(HyperText Markup Language，HTML)。HTML 的妙处在于，它能把应该如何展现文本与具体如何实现显示相分离。Berners-Lee 不仅创建一个称为超文本传输协议(HyperText Transfer Protocol，简称 HTTP)的简单协议，还同时发明了第一个 Web 浏览器，这个浏览器就叫做 World Wide Web。1990 年 11 月，第一个 Web 服务器 nxoc01.cern.ch 开始运行，Tim Berners-Lee 在自己编写的图形化 Web 浏览器“World Wide Web”上看到了最早的 Web 页面。1991 年，CERN 正式发布了 Web 技术标准。目前，与 Web 相关的各种技术标准都是由著名的W3C组织(World Wide Web Consortium)管理和维护。

1.2.2　Web 是什么

自 Web 诞生之日起，人们就没有给它下过一个精确的定义，但是我们可以通过以下的介绍来理解 Web。首先，Internet 是一个网络的网络，或者说是一个全球范围的网间网；其中分布了成千上万的计算机，它们各自扮演了不同的角色；但总的来看可以分为两种：客户机和服务器。客户机就是我们通常所使用的计算机；而服务器是一种高性能计算机，作为网络的节点，存储、处理网络上大量的数据、信息，因此也被称为网络的灵魂。此外，现在流行的所谓云，实际上就是服务器的集合。对应上面提到的各种网络服务，也可以分为邮件服务器、文件传输服务器、DNS 服务器和 Web 服务器等。

Web 服务器的作用是将本地的信息用超文本方式组织起来，方便用户在 Internet 上搜索和浏览。因此 Web 或者是 WWW，实际上是由 Internet 中被称为 Web 服务器的计算机构成的，从这个意义上来看，可以将 Web 应用看成是 Internet 应用的一个子集。

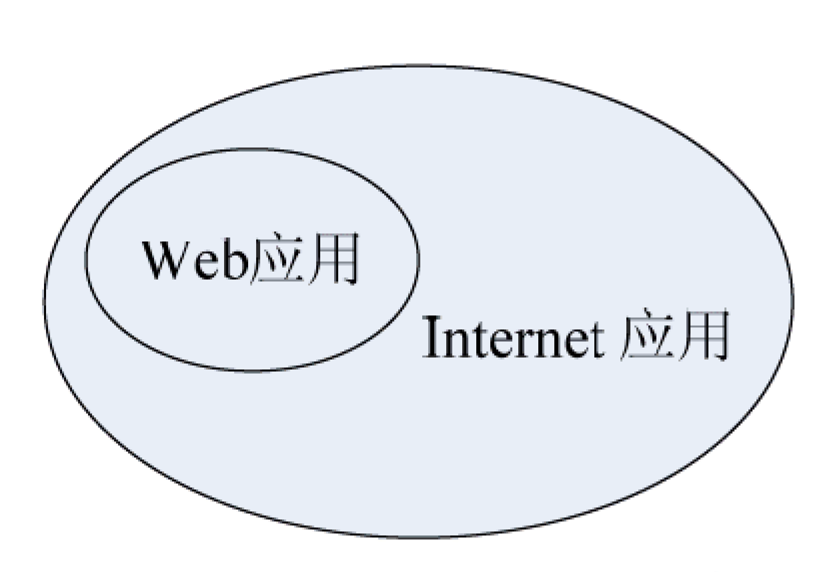


图 1-5　Internet 和 Web 的包含关系

注意：

Internet 是 Web 的基础平台，Web 是 Internet 平台上的一种应用或服务，它使人们能方便、快捷地发布和获取信息。至于这些信息是如何在 Internet 的网络层上进行传输的，对于一般的 Web 用户而言，是感觉不到的。

如果回顾一下，我们会发现在 WWW 出现伊始人们各自建立网页、互相做链接，人们上网是沿着链接冲浪。那时的 Web 是真正的“网”。但是当 Yahoo 和 Lycos 等网站建立了搜索引擎和门户站点后，人们上网的方式被改变了，人们从此到一个地方去获取所有的信息。并提出了所谓的“目标站点”模式，当人们一条条地阅读内容时，在头脑中还有一个“网”的概念吗？而这些站点在起到积极作用的同时，也控制了信息的流动并包含了过时的信息，有时还包含一些广告。

而基于 P2P 应用的出现，则把控制权重新还到用户手中。人们共享硬盘上的文件、目录甚至整个硬盘。所有人都共享了他们认为最有价值的东西，这将使互联网上信息的价值得到极大的提升。

而博客乃至于微博以及交互社区等的流行并与移动终端相结合的现实，则最大限度地将网络的应用大幅度延伸到人们的日常生活的每个角落，并通过提供这种控制权，极大地改变了内容发布的方式。

1.2.3　Web 的技术基础

从技术层面看，Web 架构的精华主要有 3 处：用统一资源定位技术(URL)实现全球资源的精确定位；用应用层协议(HTTP)实现分布式的信息传送；以超文本技术(HTML)实现信息的表现。这 3 个特点无一不与信息的分发、获取和利用有关。其实，Tim Berners-Lee 早就明确无误地告诉我们：“Web 是一个抽象的(假想的)信息空间。”也就是说，作为 Internet

上的一种应用架构，Web 的首要任务就是向人们提供信息和信息服务。

很可惜，在 Web 应用日新月异的今天，许多从事技术开发的人似乎已经忘记了 Web 架构的设计初衷。他们在自己开发的网站或 Web 应用中大肆堆砌各种所谓的“先进”技术，但最终用户能够在这些网站或应用中获得的有价值的信息却寥寥无几。这个问题绝不像评论者常说的“有路无车”或“信息匮乏”那么简单。一个 Web 开发者倘若忘记了 Web 技术的最终目标是提供信息和信息服务，他的愚蠢程度就丝毫不亚于一个在足球场上只知道卖弄技巧，却忘记了射门得分的大牌球星。从这个角度来说，评价一种 Web 开发技术优劣的标准只有一个，那就是看这种技术能否在最恰当的时间和最恰当的地点，以最恰当的方式，为最需要信息的人提供最恰当的信息服务。

Web 技术利用了一种称为超文本(Hypertext)的技术，即它使用了在文件中有着加重色的词句或图形去链接或指向其他文件、图形、声音等。它可以从一个文件中的任何一点指向另一个文件的任何一点，从而可以实现快速的信息浏览。同时超文本技术具有良好的图形用户界面，使得用户能很容易地浏览互联网中的信息。

注意：

Web 正是通过这些技术来实现其功能的，这些技术成为 Web 技术的基础。它们是从资源的定位、传输和表示等方面来帮助 Web 实现其功能的。

Web 技术中其实还包括其他更多的技术，这里介绍其中最主要的 3 个。

1. 统一资源定位技术(URL)

URL(Uniform Resource Locator，中译为“统一资源定位符”)，即通过定义资源位置的抽象标识来定位网络资源。资源被定位后，便可对其进行各种操作，例如，访问、更新、替换、查找属性等。

总体来说，URL 可按下列格式书写：

```
<scheme>:<scheme-specific-part>
```

其中，<scheme>指所用的 URL 方案名。 <scheme-specific-part>意义的解释与所用方案有关。方案名由字符组成，可包括字母(a-z)、数字(0-9)、加号(+)、句点(.)和连词符(-)，字母大小写是不分的。

对于 Internet，<scheme>指的是 Internet 协议名，目前主要包括 http、ftp、gopher、mailto、new、nntp、telnet、wais、file 等，这个列表以后还会不断扩充。

HTTP URL 方案用于表示可通过 HTTP 协议进行访问的 Internet 资源。HTTP URL 的格式如下：

```
http://<host>:<port>/<path>?<searchpart>
```

其中，<host>和<port>按标准格式，:<port>如果省略，则默认端口值为 80。<path>为 HTTP 选择器，而<searchpart>为查询字符串，它们都是可选的，如果这两项不存在，则主机或端口后的斜杠也应该省略。例如：http://www.edu.cn:80/index.asp，http 是协议，

www.edu.cn 是主机名，80 是端口号，index.asp 是要访问的资源名(此处是一个文件)。

2. 超文本标记语言(HTML)

HTML(Hyper Text Markup Language)，直译为超文本标记语言。它是一种用来制作超文本文档的简单标记语言。HTML 在诞生之初，其目的非常简单。当时 Tim Berners-Lee 将他设计的初级浏览器和编辑系统在网上合二为一，他创建了一种快速小型超文本语言来为他的这个想法服务。也设计了数十种乃至数百种未来使用的超文本格式，并想象智能客户代理通过服务器在网上进行轻松谈判并翻译文件。这同 Macintosh 的 Claris XTND 系统极为相似，不同的是它可以在任何平台和浏览器上运行。

Berners-Lee 设计的语言简易，以文本格式为基础，所以可以用任何编辑器和文字处理器来为网络创建或转换文本。并且它仅有不多的几个标签(TAG)——任何人用一个下午的时间就能掌握 HTML。网络从此迅猛发展，开启了大众在网上浏览和发布信息的时代。

超文本传输协议规定了浏览器在运行 HTML 文档时所遵循的规则和进行的操作。HTTP 协议的制定使浏览器在运行超文本时有了统一的规则和标准。用 HTML 编写的超文本文档称为 HTML 文档，它能独立于各种操作系统平台，自 1990 年以来 HTML 就一直被用作 WWW/web/万维网(World Wide Web)的信息表示语言，是全球广域网上描述网页内容和外观的标准。使用 HTML 语言描述的文件，需要通过 Web 浏览器显示出效果。HTML 包含了一对打开和关闭的标记，在当中包含有属性和值。标记描述了每个在网页上的组件，例如文本段落、文字、图形、动画、声音、表格、链接等对象。HTML 必须有特定的程序，即 Web 浏览器来完成翻译和执行的功能，通常编写者可以使用任何编辑器对 HTML 文件进行编辑，一些浏览器(如 Netscape、Internet Explorer)则提供了交互式的 HTML 编辑器。

HTML 是一种用于创建文档的标记语言，通过在文档中包含相关信息的链接来实现通过单击这个链接来访问其他文档、图像或多媒体对象，并获得关于链接项的附加信息。对于 HTML 语言更详细的介绍，将放在后面专门的章节中进行。

3. 超文本传送协议(HTTP)

HTTP(Hypertext Transfer Protocol)是一种通信协议，它允许将超文本标记语言(HTML)文档从 Web 服务器传送到 Web 浏览器。其中设计了一套相当简单的规则，用来支持超媒体系统在网络上的分布，它的出现使 Web 成为可能，如果要真正理解 Web，那么理解 HTTP 是基础。

HTTP 采用的是客户机/服务器(C/S)结构，定义了客户机/服务器之间进行“对话”的简单请求-应答规则，客户端的请求程序与运行在服务器端的接收程序建立连接，如图 1-6 所示。客户端发送请求给服务器，HTTP 规则定义了如何正确解析请求信息，服务器用应答信息回复该请求，应答信息中包含了客户端所希望得到的信息，HTTP 规则当然也定义了如何正确解析应答信息，但 HTTP 规则并没有定义网络如何建立连接、管理及信息如何在网络上发送，这些事情交给底层协议 TCP/IP 来完成，这也就是我们经常说，“Web 是站在巨人的肩膀上”的，它的真实含义是“HTTP 是建立在 TCP/IP 之上的”，HTTP 属于

应用层的协议，是 TCP/IP 的一个应用，从 TCP/IP 来看，Web(HTTP)和 TELNET、FTP、GOPHER、WAIS 等没有什么区别。

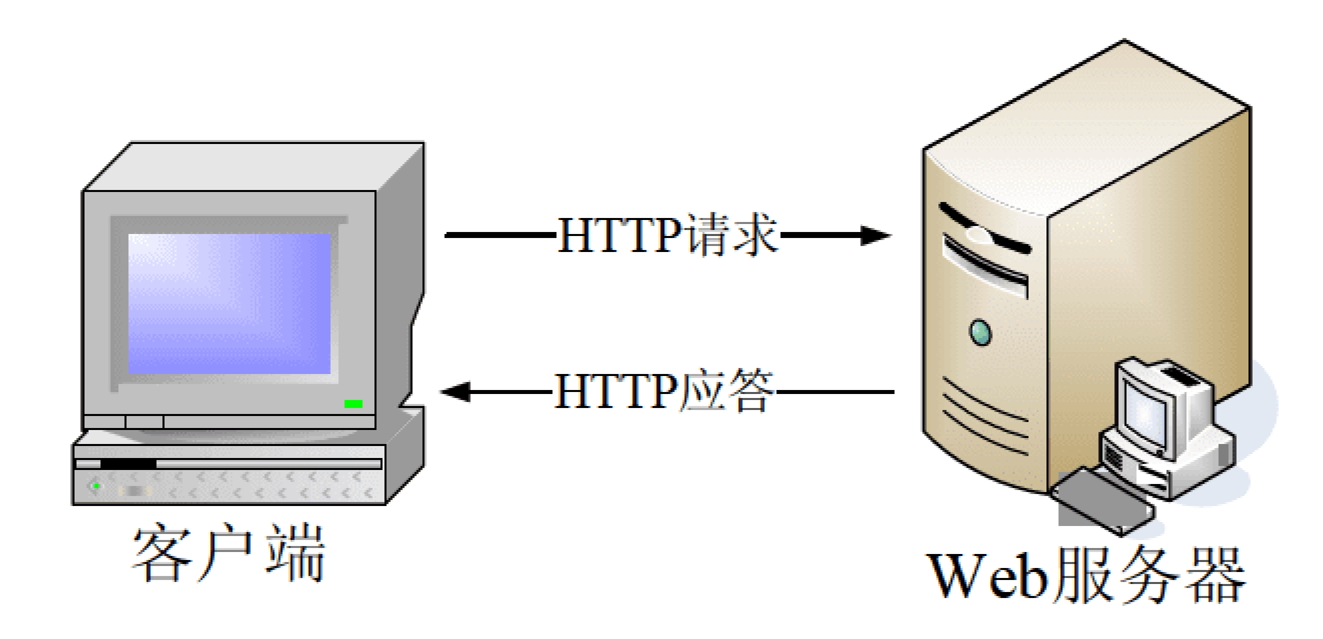


图 1-6　HTTP 的基本原理

注意：

HTTP 规则实际上定义了客户和服务器之间请求与应答的格式，使用这种规范，传输过程得以顺利完成。

4. 浏览器(Browser)

除了上面提到的三大技术外，浏览器在 Web 领域也起到了重要的作用。提到 Web 浏览器，大多数人会想到无处不在的 Microsoft Internet Explorer，直到最近像 Firefox、Safari 和 Opera 之类的浏览器日益兴起，这种情况才稍有变化。尽管许多新手可能认为 Internet Explorer 是市场上的第一个浏览器，但事实并非如此。实际上，第一个 Web 浏览器出自 Berners-Lee 之手，这是他为 NeXT 计算机创建的(这个 Web 浏览器原来取名叫 WorldWideWeb，后来改名为 Nexus)，并在 1990 年发布给 CERN 的人员。Berners-Lee 和 Jean-Francois Groff 将 WorldWideWeb 移植到 C，并把这个浏览器改名为 libwww。20 世纪 90 年代初出现了许多浏览器，包括 Nicola Pellow 编写的一个行模式浏览器(这个浏览器允许任何系统的用户都能访问 Internet，从 UNIX 到 Microsoft DOS 都涵盖在内)，还有 Samba，这是第一个面向 Macintosh 的浏览器。

1993 年 2 月，Illinois–Urbana-Champaign 大学超计算应用国家中心的 Marc Andreessen 和 Eric Bina 为 UNIX 发布了 Mosaic。几个月之后，Aleks Totic 为 Macintosh 发布了 Mosaic 的一个版本，这使得 Mosaic 成为第一个跨平台浏览器，它很快得到普及，并成为最流行的 Web 浏览器。这个技术卖给了 Spyglass，后来又归入 Microsoft 的门下，并用在 Internet Explorer 中。

1993 年，堪萨斯大学的开发人员编写了一个基于文本的浏览器，叫做 Lynx，它成为字符终端的标准。1994 年，挪威奥斯陆的一个小组开发了 Opera，1996 年这个浏览器得到了广泛使用。1994 年 12 月，Netscape 发布了 Mozilla 的 1.0 版，第一个营利性质的浏览器诞生。2002 年又发布了一个开源的版本，这发展为后来流行的 Firefox 浏览器，于 2004 年 11 月发布。

Microsoft 发布 Windows 95 时，把 Internet Explorer 1.0 作为 Microsoft Plus!包的一部分同时发布。尽管这个浏览器与操作系统集成在一起，但大多数人还是坚持使用 Netscape、Lynx 或 Opera。IE 2.0 有了很大起色，增加了对 cookie、安全套接字层(Secure Socket Layer，

SSL)和其他新兴标准的支持。这个第二版还可以用于 Macintosh，使之成为 Microsoft 的第一个跨平台浏览器。不过，大多数用户还是很执着，仍然使用他们用惯了的浏览器。

不过到了 1996 年夏天，Microsoft 发布了 3.0 版本。几乎一夜之间，人们纷纷拥向 Internet Explorer；当然，Netscape 的浏览器还是要收费，Microsoft 仍然免费提供 Internet Explore；关于浏览器领域谁主沉浮的问题，Internet 群体发生了两极分化，很多人担心 Microsoft 会像在桌面领域一样，在 Web 领域也一统天下。有些人则考虑到安全问题，而且不出所料，发布 3.0 版 9 天之后，就报告了第一个安全问题；但是到 1999 年发布 Internet Explorer 5 时，它已经逐步成为使用最广的浏览器。但根据 StatCounter 的统计，2013~2014 年全世界网民所使用的五大浏览器比例如图 1-7 所示，其中目前使用数量最多的是 Google 的 Chrome，占比达 44.4%，而 IE 占比为 22.63%。

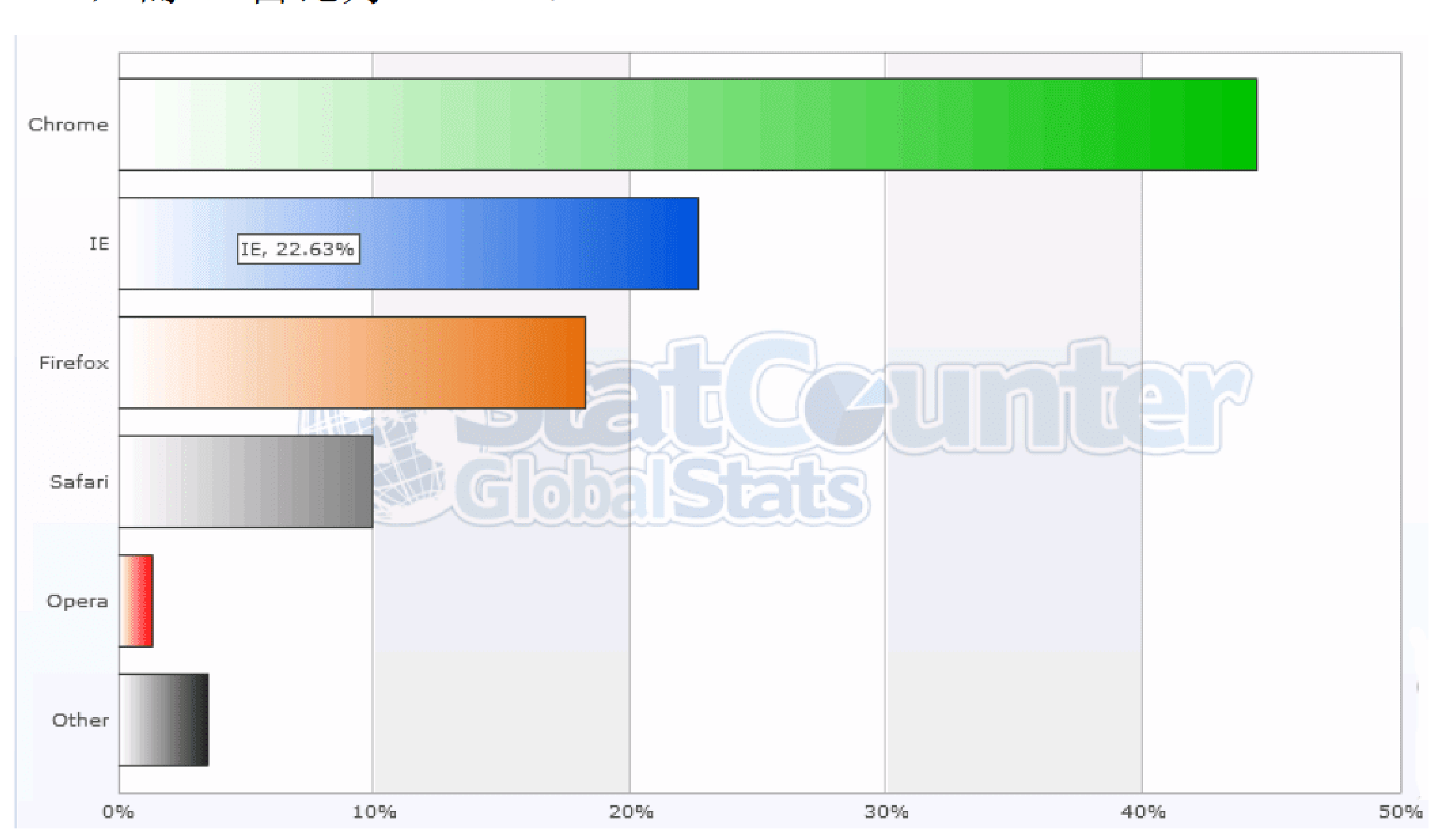


图 1-7 2013~2014 年全球五大浏览器

1.2.4 Web 的高级技术

最初，所有 Web 页面都是静态的；用户请求一个资源，服务再返回这个资源。什么都不动，也不会出现屏幕的闪动。坦率地讲，对于许多 Web 网站来说，这样确实是可以的，这些网站的 Web 页面只是数字化的文本，一旦生成，就内容固定，再发布到多处。在浏览器发展的最初阶段，Web 页面的这种静态特性是可以满足使用的；科学家们通常只是使用 Internet 来交换研究论文，大学院校也只是通过 Internet 在线发布课程信息。企业界还没有发现这个新“渠道”会提供什么商机。

实际上，刚开始时，公司主页显示的信息通常很少，无非是一些联系信息，或者只是一些文档。不过，没过多久，Web 用户就开始有新的要求，希望能得到更动态的网上体验。个人计算机成为企业的强劲后盾，从个人宿舍到住家办公室开始出现越来越多的计算机。随着人们需求的发展，用户的期望也越来越高。

1. CGI

要让 Web 更为动态，第一个办法是通用网关接口(Common Gateway Interface，简称

CGI)。与静态的 Web 获取不同，可以使用 CGI 创建程序，用户发出请求时就会执行这个程序。假设你想在 Web 网站上显示销售的商品，可以利用一个 CGI 脚本来访问商品数据库，并显示结果。通过使用简单的 HTML 表单和 CGI 脚本，可以创建简单的前台应用，这样就可以通过浏览器来购买商品。可以用多种语言编写 CGI 脚本，从 Perl 到 Visual Basic 都可以，这使得掌握不同语言的人都能编写 CGI 脚本。

不过，要创建动态的 Web 页面，CGI 并不是最安全的方法。如果采用 CGI，任何人都可以在系统上执行程序。大多数情况下这可能没有问题，但是倘若一个用户有恶意企图，很可能利用这一点，让系统运行本来不想运行的程序。尽管存在这个缺陷，但如今 CGI 仍在使用。

2. Applet

很显然，CGI 可以有所改进。1995 年 5 月，Sun 的 John Gage 和 Andreessen(目前在 Netscape 通信公司)宣布一种新的编程语言诞生，这就是 Java。Netscape Navigator 为这种新语言提供了支持，最初是为了支持机顶盒(读者可能认为，为了抢占在居室电子化方面的发展先机，最早涉足的公司是 Microsoft 和 Sony)。就像所有革命一样，Java 和 Internet 的出现恰到好处，在适当的时间、适当的地点横空出世，Java 在 Web 上发布仅几个月，就已经有数以千计的人下载 Java。由于 Netscape 的 Navigator 支持 Java，动态 Web 页面掀开了新的一页——Applet 时代到来。

Applet 允许开发人员编写小应用，这些小应用可以嵌入在 Web 页面上。只要用户使用支持 Java 的浏览器，就可以在浏览器的 Java 虚拟机(Java Virtual Machine，JVM)中运行 Applet。尽管 Applet 可以做很多事情，但它们也存在一些限制，即通常不允许读/写文件系统，不能加载本地库，而且可能无法启动客户端上的程序。除了这些限制外，Applet 还会在一个沙箱安全模型中运行，这有助于防止用户运行恶意代码。

对许多人来说，最初接触 Java 编程语言就是从 Applet 开始的，当时这是创建动态 Web 应用的一种绝好的办法。Applet 允许在浏览器中创建一个“胖”客户应用，不过必须在平台的安全限制范围内。当时，在很多领域都广泛使用了 Applet；但是，Web 群体并没有完全被 Applet“征服”。“胖”客户的开发人员都很熟悉一个问题：必须在客户端上部署适当的 Java 版本。因为 Applet 在浏览器的虚拟机中运行，所以开发人员必须确保客户端安装了适当版本的 Java。尽管这个问题并非无法解决，但确实妨碍了 Applet 技术的进一步推广。而且如果 Applet 写得不好，很可能对客户主机造成影响，这使许多客户对于是否采用基于 Applet 的解决方案犹豫不定。

3. JavaScript

Netscape 创建了一种脚本语言，并最终称之为 JavaScript(建立原型时本来叫做 Mocha，正式发布之前曾经改名为 LiveWire 和 LiveScript，不过最后终于确定为 JavaScript)。设计 JavaScript 是为了让不太熟悉 Java 的网页设计人员和程序员能够更轻松地开发 Applet(当然，Microsoft 也推出了与 JavaScript 相对应的脚本语言，称为 VBScript)。Netscape 请 Brendan

Eich 来设计和实现这种新语言，Brendan Eich 认为在这种情况下需要的是一种动态类型脚本语言。由于缺乏开发工具，缺少有用的错误消息和调试工具，JavaScript 很受非议。尽管如此，JavaScript 也仍然不失为是一种创建动态 Web 应用的强大方法。

JavaScript 是一种基于对象和事件驱动并具有安全性能的脚本语言，有了 JavaScript，可使网页变得生动。使用它的目的是与 HTML 超文本标识语言、Java 脚本语言一起实现在一个网页中链接多个对象，与网络客户交互作用，从而可以开发客户端的应用程序。它是通过嵌入或调入在标准的 HTML 语言中实现的。

最初，创建 JavaScript 是为了帮助开发人员动态地修改页面上的标记，以便为客户提供更丰富的体验。人们越来越认识到，页面也可以当作对象，因此文档对象模型(Document Object Model，DOM)应运而生。刚开始，JavaScript 和 DOM 紧密地交织在一起，但最后它们还是“分道扬镳”，并各自发展。DOM 是页面的一个面向对象模型，可以用某种脚本语言(如 JavaScript 或 VBScript)进行修改。关于这部分内容的详细介绍读者可以查阅后面章节的内容。

最后，万维网协会(World Wide Web Consortium，简称 W3C)介入，完成了 DOM 的标准化，而欧洲计算机制造商协会(European Computer Manufacturers Association，ECMA)则批准了将 JavaScript 作为 ECMAScript 规范。根据这些标准编写的页面和脚本在遵循相应原则的任何浏览器上都应该有相同的外观和表现。

在最初的几年中，JavaScript 的发展比较坎坷，这是许多因素造成的。首先，浏览器支持很不一致(即使是今天，同样的脚本在不同浏览器上也可能有不同的表现)，而且客户可以自由地把 JavaScript 关闭(由于存在一些已知的安全漏洞，因此往往鼓励用户把 JavaScript 关掉)。由于开发 JavaScript 有一定难度，且使用 JavaScript 完成的代码是对用户公开的，这使得许多开发人员退避三舍，很少使用这种语言，有些开发人员干脆不考虑 JavaScript，认为这是图形设计人员使用的一种“玩具”语言。许多人曾试图使用、测试和调试复杂的 JavaScript，并为此身心俱疲，所以大多数人在经历了这种痛苦之后，最终还是满足于创建简单的基于表单的应用。

4. Servlet、JSP、ASP 和 PHP 等

尽管 Applet 是基于 Web 的，但“胖”客户应用存在的许多问题在 Applet 身上也有所体现。在大量使用拨号连接的年代，要下载一个复杂 Applet 的完整代码，可能要花很多时间，这往往不是用户所能忍受的。开发人员还要考虑客户端上的 Java 版本，有些虚拟机还有更多的要求。理想情况下只需提供静态的 Web 页面，毕竟，这正是设计 Internet 的本来目的。当然，尽管静态页面是静态的，但是如果能在服务器上动态地生成内容，再把静态的内容返回，这就更好了。

在 Java 问世一年左右，Sun 引入了 Servlet。Java 代码不用像 Applet 那样在客户端浏览器中运行；它在一个应用服务器上运行。这样，开发人员就能充分利用现有的业务应用，而且，如果需要升级为最新的 Java 版本，只需要考虑服务器端的升级就行了。正如 Java 所推崇的“一次编写，到处运行”，这一点使得开发人员可以选择最先进的应用服务器和

服务器环境，这也是这种新技术的另一个优点。如此，Servlet 就可以取代 CGI 脚本了。

Servlet 是向前迈出的很大一步，它提供了对整个 Java 应用编程接口(API)的完全访问，而且提供了一个完备的库可以处理 HTTP。不过，Servlet 并不是十全十美的，使用 Servlet 来设计界面可能很困难。在一个典型的 Servlet 交互中，先要从用户得到一些信息，完成某种业务逻辑，然后使用一些“打印行”创建 HTML，为用户显示结果。以下是一个简单的 Servlet 代码片段。

```
response.setContentType("text/html;charset=UTF-8");
PrintWriter out = response.getWriter();
out.println("<html>");
out.println("<head>");
out.println("<title>Servlet SimpleServlet</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1>Hello World</h1>");
out.println("<p>Imagine if this were more complex.</p>");
out.println("</body>");
out.println("</html>");
out.close();
```

Servlet 不仅容易出错，很难生成可视化显示，而且还无法做到人尽其才。一般来说，编写服务器端代码的人往往是软件开发人员，由于只是对算法和编译器很精通，他们并不能设计出精美网站的图形和页面布局。使用这种模式进行开发，业务开发人员不仅要编写业务逻辑，还必须考虑怎样创建一致的设计。因此，很有必要将表示与业务逻辑分离，其实这里需要的就是 Java Server Pages(JSP)。

在某种程度上，JSP 是对 Microsoft 的 Active Server Pages(ASP)做出的一个回应。Microsoft 从 Sun 在 Servlet 规范上所犯的错误吸取了教训，并创建了 ASP 来简化动态页面的开发。Microsoft 增加了一些工具来支持，并与其 Web 服务器紧密集成。JSP 和 ASP 都具有将业务处理与表示布局相分离的特征，从这个意义上讲，二者是相似的。虽然存在一些技术上的差别(Sun 也从 Microsoft 那里学到了教训)，但它们有一个最大的共同点，即都允许 Web 设计人员能够把重点放在布局上，而软件开发人员可以集中开发业务逻辑。以下代码展示了一段简单的 JSP 源码。

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN""http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Hello World</title>
</head>
<body>
```

```
<h1>Hello World</h1>
<p>This code is more familiar for Web developers.</p>
</body>
</html>
```

当然，Microsoft 和 Sun 并没有垄断服务器端解决方案。除了 JSP 和 ASP 以外还有许多其他的方案，如 PHP、ColdFusion 等。有些开发人员喜欢独特的工具，而有一些开发人员则倾向于更简单的语言。从目前来看，所有这些解决方案完成的目标都是一样的，它们都是要动态生成 HTML。

5. Flash

并不是只有 Microsoft 和 Sun 在努力寻找办法来解决动态 Web 页面问题。1996 年夏天，FutureWave 发布了一个名叫 FutureSplash Animator 的产品。这个产品起源于一个基于 Java 的动画播放器，FutureWave 很快被 Macromedia 兼并，Macromedia 则将这个产品改名为 Flash。

Flash 是交互式矢量图和 Web 动画的标准。网页设计者使用 Flash 创作出既漂亮又可改变尺寸的导航界面以及其他奇特的效果。Flash 通常也指 Macromedia Flash Player(现称为 Adobe Flash Player)。2012 年 8 月 15 日，Flash 退出 Android 平台，正式告别移动端。Adobe Flash 最新版本也宣布支持 3D，这将会是 Flash 未来发展的趋势，也会是网页游戏的主流技术。

利用 Flash，设计人员可以创建令人惊叹的动态应用，它可以在 Web 上发布高度交互性的应用，几乎与"胖"客户应用相差无几。但是不同于 Applet、Servlet 和 CGI 脚本，Flash 不需要编程技巧，很容易上手，在 20 世纪 90 年代末期，掌握 Flash 是一个很重要的特长，因为许多老板都非常需要有这种技能的员工。不过，这种易用性也是有代价的(所谓的胖客户应用在本章的后面将有较为详细的介绍)。

像许多解决方案一样，Flash 需要客户端软件支持。尽管许多流行的操作系统和浏览器上都内置有其所需的播放器插件，但并非所有的浏览器都有。虽然能免费下载，但由于担心同时携带病毒，使得许多用户拒绝安装这个软件，这一点限制了此解决方案的通用性。在某些情况下，Flash 应用可能还需要较大的网络带宽才能很好地工作，这也限制了 Flash 的推广(因此产生了某些页面上出现的所谓"跳过动画"的链接)。尽管有些网站选择建立多个版本的 Web 应用，分别适应于不同的连接速度，但是许多公司都无法承受支持两个或更多网站所增加的开发开销。

注意：

创建 Flash 应用需要专用的软件和浏览器插件；而 Applet 可以用文本编辑器编写，而且有一个免费的 Java 开发包(Java Development Kit，JDK)，Flash 则不同，使用完整的 Flash 工具包需要支付较高的费用。尽管这些因素不是难以逾越的障碍，但它们确实减慢了 Flash 在动态 Web 应用道路上的前进脚步。

6. SilverLight

Microsoft SilverLight 中文名称为“微软银光”，是微软所发展的 Web 前端应用程序开发解决方案，亦是微软丰富型互联网应用程序(Rich Internet Application)策略的主要应用程序开发平台之一。其以浏览器的外挂组件方式，提供 Web 应用程序中多媒体(含影音流与音效流)与高度交互性前端应用程序的解决方案，同时它也是微软 UX(用户经验)策略中的一环，更是微软试图将美术设计和程序开发人员的工作明确切分与协同合作发展应用程序的尝试之一。

对于互联网用户来说，SilverLight 是一个安装简单的插件程序。用户只要安装了这个插件程序，就可以在 Windows 和 Macintosh 等操作系统的多种浏览器中运行相应版本的 SilverLight 应用程序，享受视频分享、在线游戏、广告动画、交互丰富的网络服务等。

对于开发设计人员而言，SilverLight 是一种融合了微软的多种技术的 Web 呈现技术。它提供了一套开发框架，并通过使用基于向量的图像、图层技术，支持任何尺寸图像的无缝整合，对基于 ASP .NET、AJAX 在内的 Web 开发环境实现了无缝连接。Silverlight 使开发设计人员能够更好地协作，有效地创造出能在 Windows 和 Macintosh 上应用的多种浏览器。

简而言之，SilverLight 是一个跨浏览器、跨平台的插件，为网络带来下一代基于.NET 媒体体验和丰富的交互式应用程序。对运行在 Macintosh 和 Windows 上的主流浏览器，SilverLight 提供了统一而丰富的用户体验。通过 SilverLight 这个小小的浏览器插件，视频、交互性内容，以及其他应用能完好地融合在一起。

1.2.5　WWW 的扩展

WWW 发展迅猛，将来许多新的技术会带来革命性的进步，以下是一些新的方向。

1. DHTML 革命

Microsoft 和 Netscape 发布其各自浏览器的第 4 版时，Web 开发人员有了一个新的选择，开发了动态 HTML(Dynamic HTML，简称为 DHTML)技术。有些人可能认为 DHTML 不是一个 W3C 标准；它更像是一种销售手段。实际上，DHTML 结合了 HTML、层联样式表(Cascading Style Sheets，简称 CSS)、JavaScript 和 DOM。这些技术的结合使得开发人员可以动态地修改 Web 页面的内容和结构。

最初对 DHTML 的反响很好。不过，它需要的浏览器版本还没有得到广泛应用。尽管 Internet Explorer 和 Netscape 都支持 DHTML，但是它们的实现大相径庭，这说明开发人员必须知道他们的客户使用什么浏览器。而这通常意味着，需要大量代码来检查浏览器的类型和版本，这就进一步增加了开发的开销。有些人对于尝试这种方法很是迟疑，因为 DHTML 还没有一个官方的标准。不过，应该相信将来一定会更好。

2. XML 技术

20 世纪 90 年代中期，基于 SGML，衍生出了 W3C 的可扩展标记语言(eXtensible Markup

Language，XML)，自此以后，XML 变得极为流行。许多人把 XML 视为解决所有计算机开发问题的灵丹妙药，以至于 XML 几乎无处不在。实际上，Microsoft 早已经宣布，将来的 Office 将支持 XML 文件格式。

如今，我们至少有 4 种 XML 衍生语言可以创建 Web 应用(W3C 的 XHTML 不包括在内)，分别是：Mozilla 的 XUL ；XAMJ，这是结合 Java 的一种开源语言；Macromedia 的 MXML；以及 Microsoft 的 XAML。现分别对这 4 种语言详细介绍如下。

- XUL：XUL(拼作“zool”)代表 XML 用户接口语言(XML User Interface Language)，由 Mozilla Foundation 推出。流行的 Firefox 浏览器和 Thunderbird 邮件客户都是用 XUL 编写的。利用 XUL，开发人员能构建功能很丰富的应用，可以与 Internet 连接，也可以没有连接。为了让熟悉 DHTML 的开发人员尽快地学会，XUL 设计为可以为诸如窗口和按钮等标准界面部件提供跨平台支持。虽然它本身不是一个标准，但 XUL 所基于的都是标准，如 HTML 4.0、CSS、 DOM、XML 和 ECMAScript 等。XUL 应用可以在浏览器上运行，也可以安装在一个客户主机上。当然，XUL 也不是没有缺点。XUL 需要 Gecko 引擎，而且目前 Internet Explorer 还没有相应的插件。尽管 Firefox 在浏览器市场已经有了一定的份额，但少了 Internet Explorer 的支持还是受到了很大影响，这使得大多数应用都无法使用 XUL。目前开展的很多项目都是力图在多个平台上使用 XUL，包括 Eclipse。
- XAML：XAML (拼作“zammel”)是 Microsoft 推出的 Vista 操作系统的一个组件。XAML 是可扩展应用标记语言(eXtensible Application Markup Language)的缩写，它为使用 Vista 创建用户界面定义了一个标准。与 HTML 类似，XAML 使用标签来创建标准元素，如按钮和文本框等。XAML 建立在 Microsoft 的 .NET 平台之上，而且可以编译为.NET 类。应当很清楚 XAML 的局限所在，其作为一个 Microsoft 产品，这就要求必须使用 Microsoft 的操作系统。在许多情况下，这可能不成问题，但是有些公司使用的不是 Microsoft 的操作系统，总不能削足适履吧。Vista 交付的日期不断推迟的过程中，XAML 也有了很大变化，其他不再只是一个播放器，据说，在未来几年内，我们可能会看到一个全新的 XAML。
- MXML：Macromedia 创建了 MXML，作为与其 Flex 技术一同使用的一种标记语言；MXML 设计为与 HTML 很相似，可以以一种声明的方式来设计界面。与 XUL 和 XAML 类似，MXML 提供了更丰富的界面组件，如 DataGrid 和 TabNavigator，利用这些组件可以创建功能丰富的 Internet 应用。不过，MXML 不能独立使用；它依赖于 Flex 和 ActionScript 编程语言来编写业务逻辑。MXML 与 Flash 有同样的一些限制，表现为，它是专用的，而且依赖于价格昂贵的开发和部署环境。尽管将来.NET 可能会对 MXML 提供支持，但现在 Flex 只能在 Java 2 企业版(Java 2 Enterprise Edition，简称为 J2EE)应用服务器上运行，如 Tomcat 和 IBM 的 WebSphere，这就进一步限制了 MXML 的广泛采用。
- XAMJ：让人欣喜的是，开源群体又向有关界面设计的 XML 衍生语言世界增加了新的成员。XAMJ 作为另一种跨平台的语言，为 Web 应用开发人员又提供了一个

工具。这种衍生语言基于 Java，Java 是当前最流行的面向对象语言之一，XAMJ 也因此获得了面向对象语言的强大功能。XAMJ 实际上想要替代基于 XAML 或 HTML 的应用，力图寻找一种更为安全的方法，既不依赖于某种特定的框架，也不需要高速的 Internet 连接。XAMJ 是一种编译型语言，建立在“Clientlet”体系结构之上，尽管基于 XAMJ 的程序也可以是独立的应用，但一般来讲都是基于 Web 的应用。

谈到“以 X 开头的东西”时，是一定要涉及 W3C XForms 规范的。XForms 设计为支持一种更丰富的用户界面，而且能够将数据与表示解耦合。毋庸置疑，XForms 数据是 XML，这样就能使用现有的 XML 技术，如 XPath 和 XML Schema。标准 HTML 能做的，XForms 都能做，而且 XForms 还有更多功能，包括动态检查阈值、与 Web 服务集成等。不同于其他的许多 W3C 规范，XForms 不需要新的浏览器，可以使用现在已有的许多浏览器实现。与大多数 XML 衍生语言一样，XForms 是一种全新的方法，所以对于这种方法何时得到采纳，目前还不能确定。

注意：

XML 技术正在快速的进步中，目前，很多应用只是将 XML 作为一种数据交换或数据存储的手段，其实 XML 的功能远不止这些。

3. Ajax 技术

Ajax(异步 JavaScript 和 XML)是个新产生的术语，它从两方面提供强大的性能。这两个特性在多年来一直被网络开发者所忽略，直到最近 Gmail、Google suggest 和 Google Maps 的横空出世才使人们开始意识到其重要性。这两项是：

- 无须重新装载整个页面便能向服务器发送请求；
- 对 XML 文档的解析和处理。

Ajax 描述了一组技术，它使浏览器可以为用户提供更为自然的浏览体验。在 Ajax 之前，Web 站点强制用户进入提交/等待/重新显示范例，用户的动作总是与服务器的“思考时间”同步。Ajax 提供与服务器异步通信的能力，从而使用户从请求/响应的循环中解脱出来。借助于 Ajax，可以在用户单击按钮时，使用 JavaScript 和 DHTML 立即更新 UI，并向服务器发出异步请求，以执行更新或查询数据库。当请求返回时，就可以使用 JavaScript 和 CSS 来相应地更新 UI，而不是刷新整个页面。最重要的是，用户甚至不知道浏览器正在与服务器通信：Web 站点看起来是即时响应的。

Ajax 是一项非常有前途的技术，本书将在第 8 章的第 8.3 节介绍这项技术。

1.3　Web 应用开发的需求与方法

最初，Web 应用技术与应用程序的开发技术是独立发展的。在各种日益复杂的应用需

求推动下，两者日益融合，但又各具特色。目前基于 Web 的应用已经成为一种主流的解决方案。下面就从 Web 和应用程序发展的角度，来分析 Web 运用开发的需求与方法。

1.3.1　Web 应用的需求

自从 Web 诞生以来，经过十几年的发展，Web 应用的架构经历了静态页面、活动页面以及动态页面的转变发展等。

1. 静态页面

静态页面是存储于服务器的文件，其内容在生成该文档时就已定义好，并且始终不变。Web 上最早的内容就是这种形式的，这些页面中可以包含多种媒体元素，信息资源的表现形式也是多样化的，而且页面间可以通过超链接进行关联，便于用户浏览、检索。典型的静态页面在访问时可以看到其 URL 中的资源的扩展名通常为 html，如图 1-8 所示。

静态页面的内容是编著者创作页面时确定好的，一旦将所有内容保存在服务器中并发布，任何人去访问都可以看到相同的内容。虽然静态页面已经可以为用户提供通过远程来访问信息资源的良好途径，但这种架构方式实际上存在很大的局限性。对于信息资源的用户而言，是不能和页面进行交互的，页面的内容也不会因为用户所做的操作而发生变化。对于信息资源的提供者而言，静态页面必须手工制作，手工更新和发布，其开发难度不大。

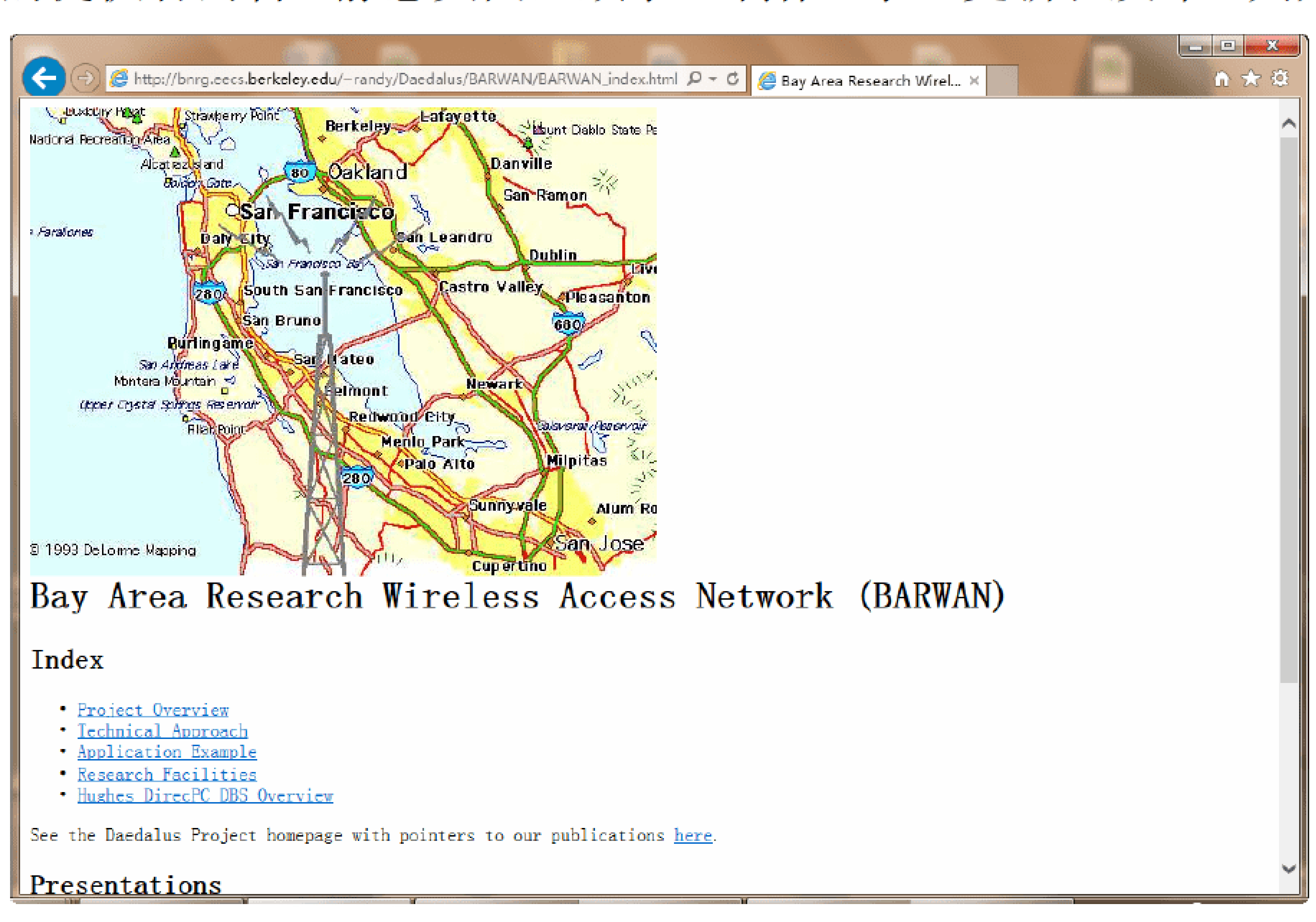


图 1-8　静态页面

2. 动态页面

对于类似网上购物、股市行情等需要实时更新的信息而言，静态页面是难以胜任的。同时静态页面也无法实现显示形式、内容等的个性化定制。动态页面就是针对这些问题应运而生的。

动态页面是在浏览器访问 Web 服务器时，由 Web 服务器创建的。当浏览器向服务器发出请求时，Web 服务器运行一个应用程序，创建动态文档，并返回给浏览器作为应答。因此，不同用户在不同时刻访问同一个动态页面时，可能就会得到不同的结果，从这个角

度来看，动态页面的内容是变化的，如图 1-9 所示。Web 服务器还可以将数据库中的最新数据返回给用户。但是动态页面需要由服务器实时生成，服务器的负荷较前面提到的静态方式要大；同时，其开发难度也较大，这对开发人员提出了更高的要求。

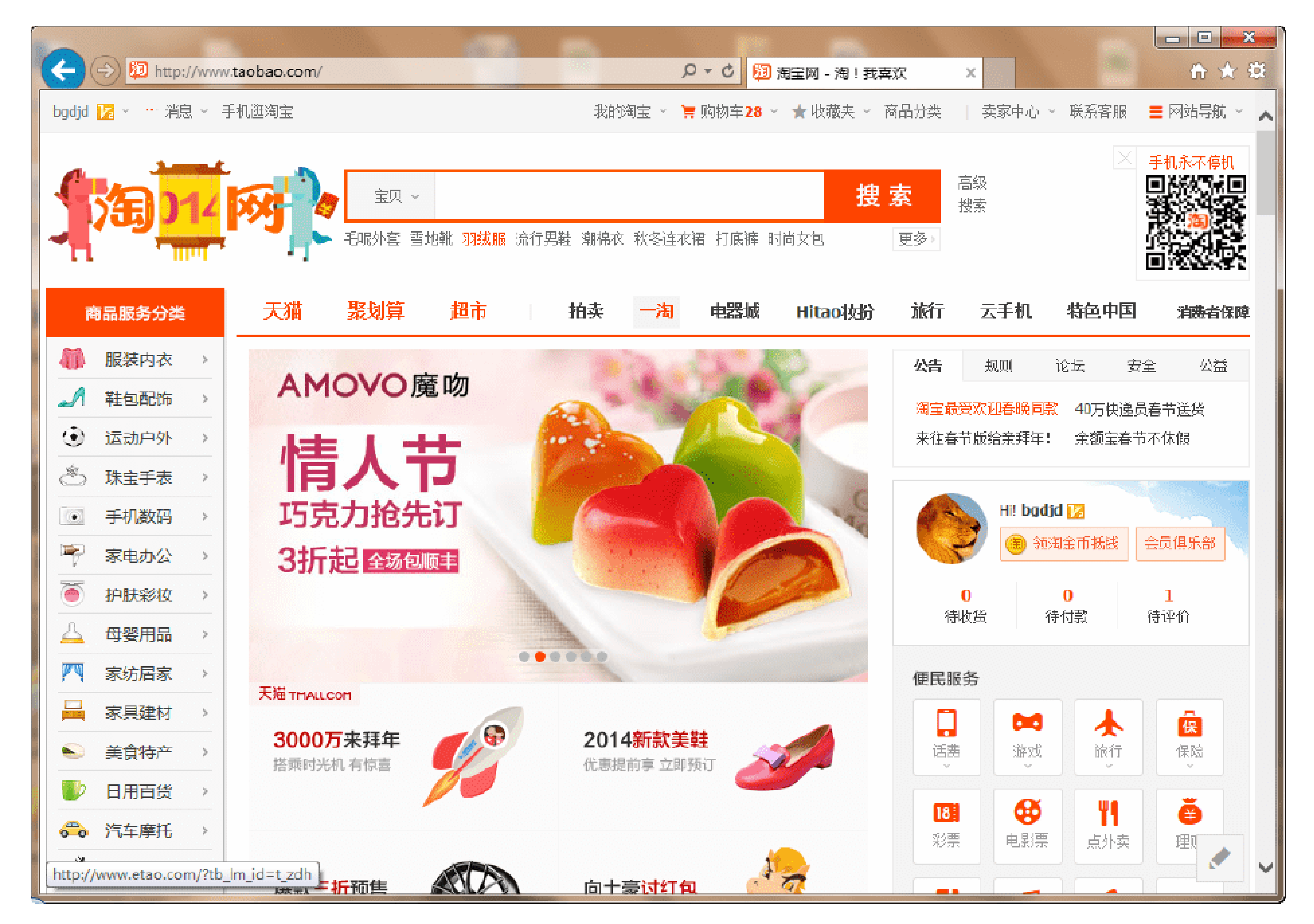


图 1-9　动态页面

3. 活动页面

提出活动页面是为了在保持动态页面优点的基础上，又能避免服务器负担过重的问题。即在传统 HTML 文档的基础上，加入诸如 Java Applet、VBScript 脚本、ActiveX 控件、Flash 插件等活动元素。首先，由服务器提供 HTML 文档和相关的活动元素，它们经客户端下载后在客户端运行，浏览器执行这些活动程序后再获得所需的信息，因此所显示的内容并不完全由服务器产生。用户通过这些元素可以和 Web 服务器进行交互，只要用户程序在运行，该页面就可不断变化保持最新，如图 1-10 所示。

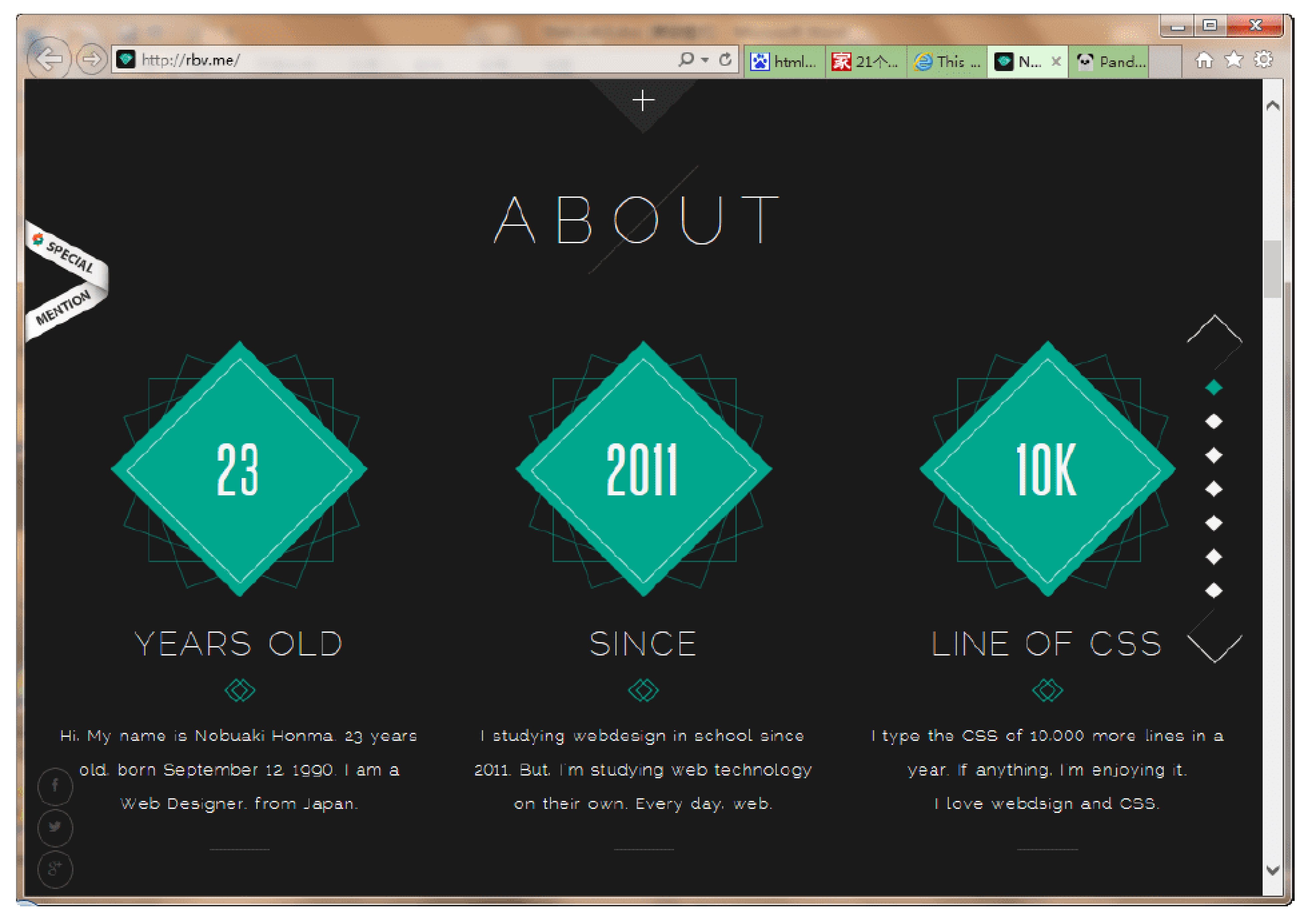


图 1-10　活动页面

由于这些元素在客户端运行，所以可以实现快速的响应和显示，但它们对客户端计算机的硬件配置和浏览器软件提出了一定的要求。此外，实现活动页面也需要一系列新技术的支持，包括 ActiveX、Java Applet 和 Flash 插件等。

从目前的情况来看，Internet 市场仍具有巨大的发展潜力，未来其应用将涵盖从办公室共享信息到市场营销、服务等广泛领域。另外，Internet 带来的电子贸易正改变着现今商业活动的传统模式，其提供的方便而广泛的互连必将对未来社会生活的各个方面带来影响。

1.3.2　应用程序发展的需求

从应用程序开发模式发展的角度来看，从最早的单机应用，到后来的 C/S 模式(客户/服务器模式)，再到现在流行的 B/S 模式(Browser/Server)、SOA、云计算等，是由简单的两层结构逐步演变为三层甚至是多层的。此外，RIA、分布式应用、设计模式和各种高级的架构模式等也在需求的推动下得到日益广泛的应用。

1. 两层结构(Two-Tier)

所谓的两层结构指的是客户机、服务器，即 C/S 结构，其结构如图 1-11 所示。通常来说，数据库位于服务器端，而客户端应用提供了与用户接口的界面，同时还包含了对服务器上的数据进行操作的一系列规则(商业逻辑)。在这种模式下，服务器仅仅需要承担数据访问的任务，而客户端程序不仅需要完成业务逻辑，即数据处理的任务，还需要负责数据的显示形式，即展示问题。

注意：

通常将 C/S 这种模式两层结构的部署方式形象地称为“胖客户机/瘦服务器”(Fat Client/Thin Server)。

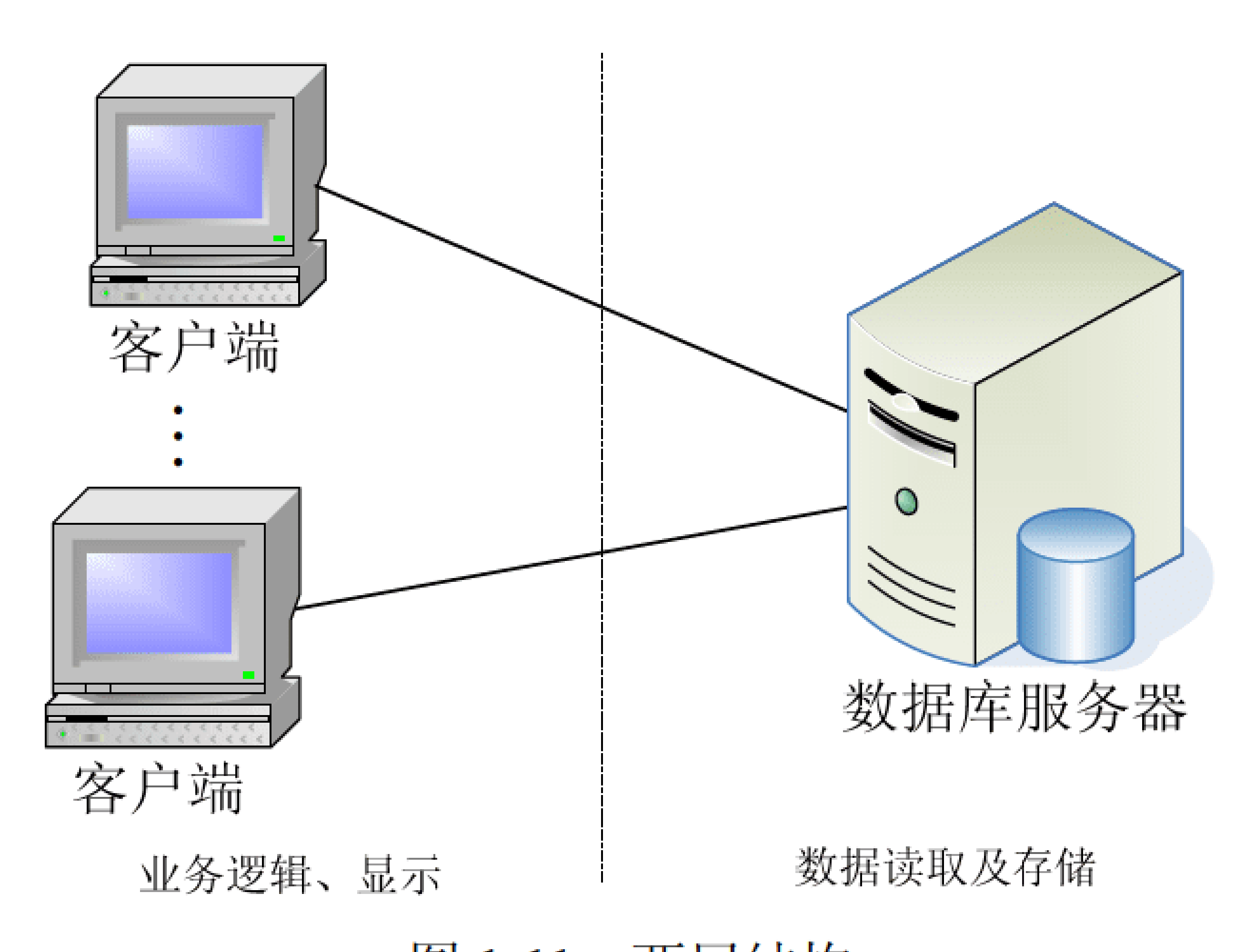


图 1-11　两层结构

客户机/服务器使得多个客户可以同时访问服务器上的数据库。但是，两层结构也有不足之处。在这种结构中，所有的数据处理规则都与某个应用程序相关联。一旦业务逻辑发生变化，必须重新修改和发布客户端的应用程序。如果客户的数量巨大，这个工作将变得十分繁重和费时。因此，两层模式难以适应大规模分布式的应用需求。

2. 三层结构(Three-Tier)

三层结构就是为了解决两层结构所存在的问题的。从功能的角度，将整个应用的功能分成表示层、功能层和数据层三部分，其结构如图 1-12 所示。其解决方案是，对这三层进行明确分割，并在逻辑上使其独立。原来的数据层作为 DBMS 已经独立出来，所以关键是要将表示层和功能层分离成各自独立的程序，并且还要使这两层间的接口简洁明了。

一般情况是只将表示层配置在客户机中，与二层 C/S 结构相比，其程序的可维护性要好得多，但是其他问题并未得到解决。客户机的负荷较重，其业务处理所需的数据要从服务器传给客户机，所以系统的性能容易变坏。如果将功能层和数据层分别放在不同的服务器中，如图 1-12 所示，则服务器和服务器之间也要进行数据传送。但是，由于在这种形态中三层是分别放在各自不同的硬件系统上的，所以灵活性很高，能够适应客户机数目的增加和负荷的变动。例如，在追加新业务处理时，可以相应增加功能层服务器的数量。因此，系统规模越大，这种形态的优点就越显著。

注意：

由于服务器承担了大部分的处理工作，因此常常将这种模式称为“瘦客户机/胖服务器”(Thin Client/ Fat Server)。

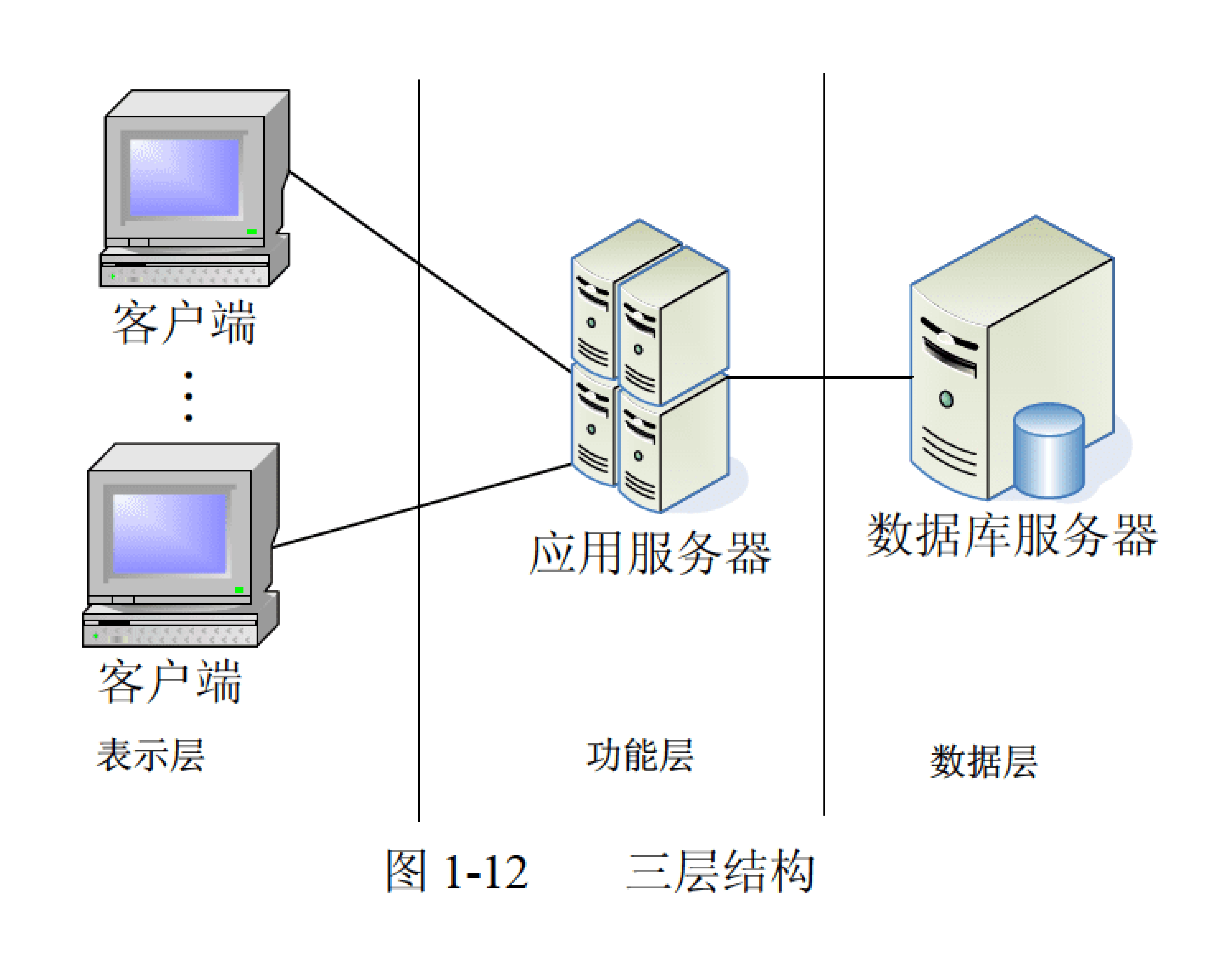


图 1-12　三层结构

3. 基于 Web 的 B/S 模式

随着 Web 的广泛运用，人们发现在某些情况下可以使用 Web 来取代以往的应用程序。此时，将 Web 浏览器作为表示层；Web 服务器上的各种服务器端应用程序来充当功能层；而数据层使用数据库服务器。为了与传统的三层结构相区别，将它称为 B/S 模式。以下对此进行分析。

(1) 静态模式

它的服务器端基本上只由 Web 服务器构成，它要发布的内容以文件的形式保存在 Web 服务器上，只能通过 HTML 文件提供静态的 Web 内容，所有的服务内容必须预先定义编辑好，其结构如图 1-13 所示。用户可以通过 URL 直接定位到这些定制好的 HTML 文件进行存取，这一模式比较简单，并且可靠性比较高，实现起来也比较容易，但是提供的内容

比较单调，且时效性及可维护性均较差，现在较大的网站已很少采用。

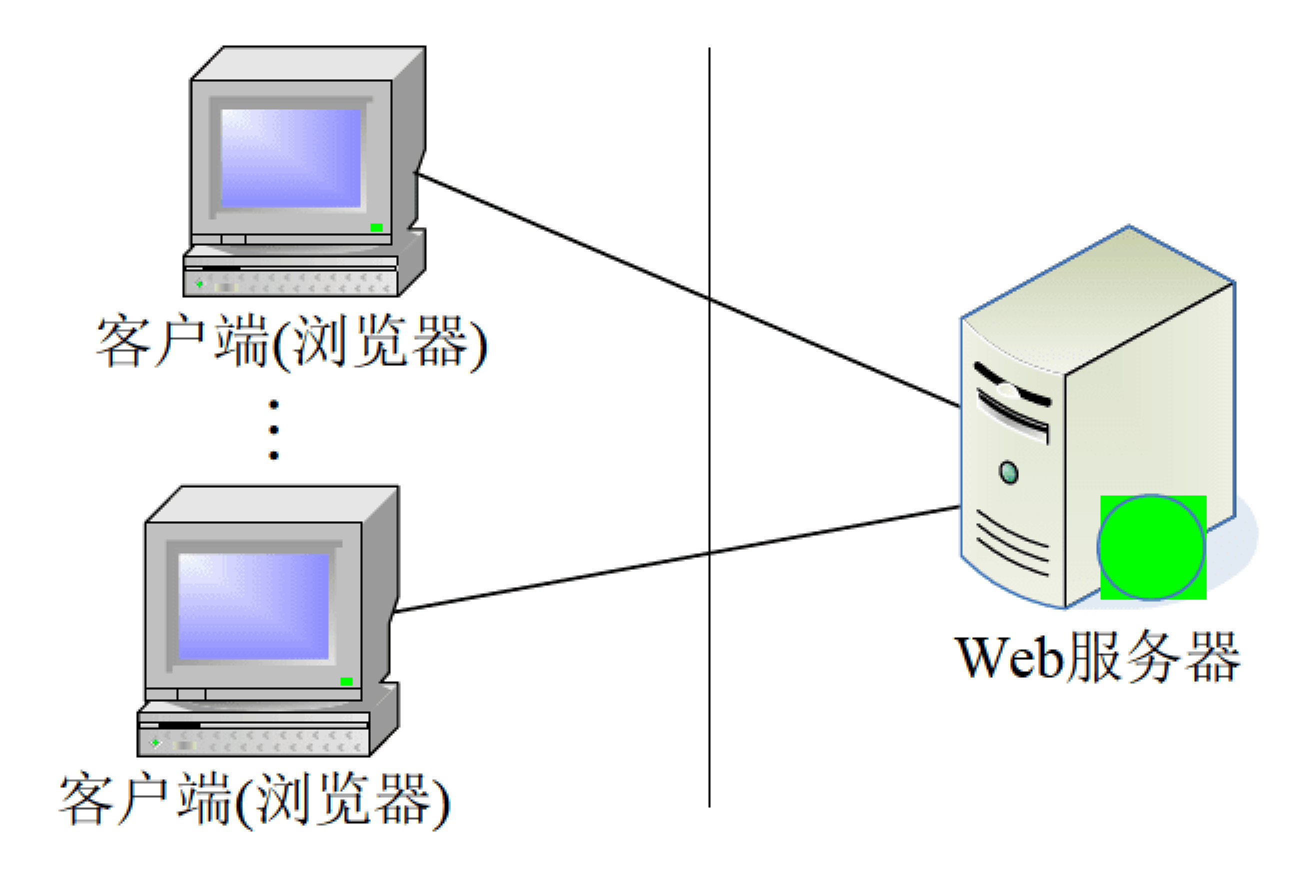


图 1-13　静态模式

(2) 一般动态模式

一般动态模式是在当前应用比较多的一种结构模式。这种模式在服务器端增加了一台数据库服务器，其结构如图 1-14 所示。它可以给用户提供动态的信息服务，通过定制页面模板，添加到后台数据库的信息可即时发布到请求的客户，保证了信息的时效性。但由于它增加了 Web 服务器的负担，因此降低了 Web 服务器的稳定性。具体的实现方式大致上可通过 ASP 脚本语言、PHP 脚本语言、普通的 CGI 程序或 ISAPI 及 NSAPI 等来实现。

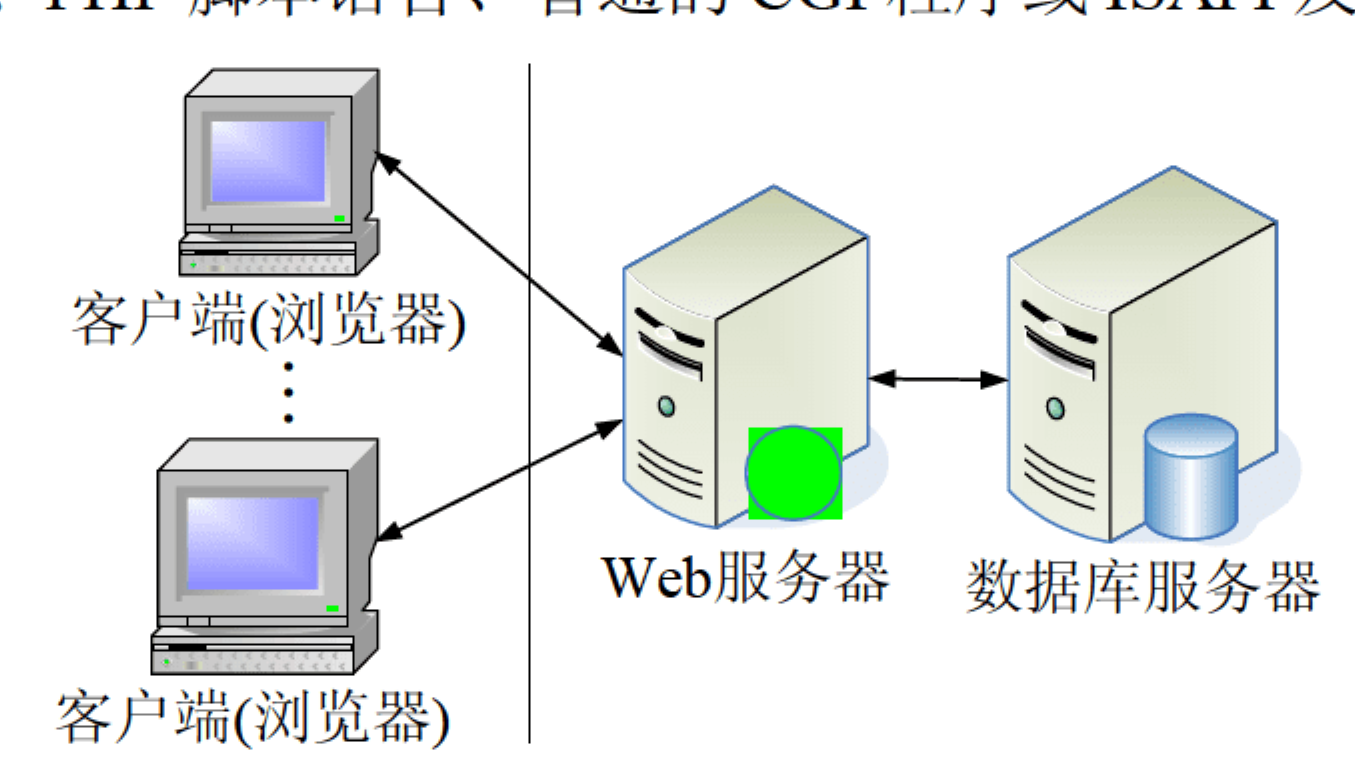


图 1-14　一般动态模式

(3) 多层动态模式

多层动态模式是在 Web 服务器和后台数据库服务器之间增加了一层应用服务器，其结构如图 1-15 所示。这是一种先进的结构模式，在国外的一些大型知名网站有所应用，像 Microsoft 的站点以及国外的一些大型电子商务站点均采用了这种结构模式。由于将一些复杂的企业逻辑及数据库的连接服务等封装到中间层上，所以减轻了 Web 服务器的负担。

具有负载平衡与容错的功能。可以通过各种技术来实现，比如，通过 ASP 脚本结合 COM/COM＋或者是 CGI 或 ISAPI 结合 COM / COM＋、PHP 脚本结合 CORBA 构件技术来实现。大多数构件均是已经编译的可执行代码，在执行速度上要比单纯的脚本语言快得多。这种结构属于典型的分布式 Web 应用系统。

B/S 模式的最大优势在于将应用程序部署在 Web 上，能创建跨平台的应用，避免多次创建和分发同一个软件的多个版本。服务器端的应用程序使用 Web 服务器上生成的 HTML 文档，这样可以被几乎是所有平台上的用户浏览。

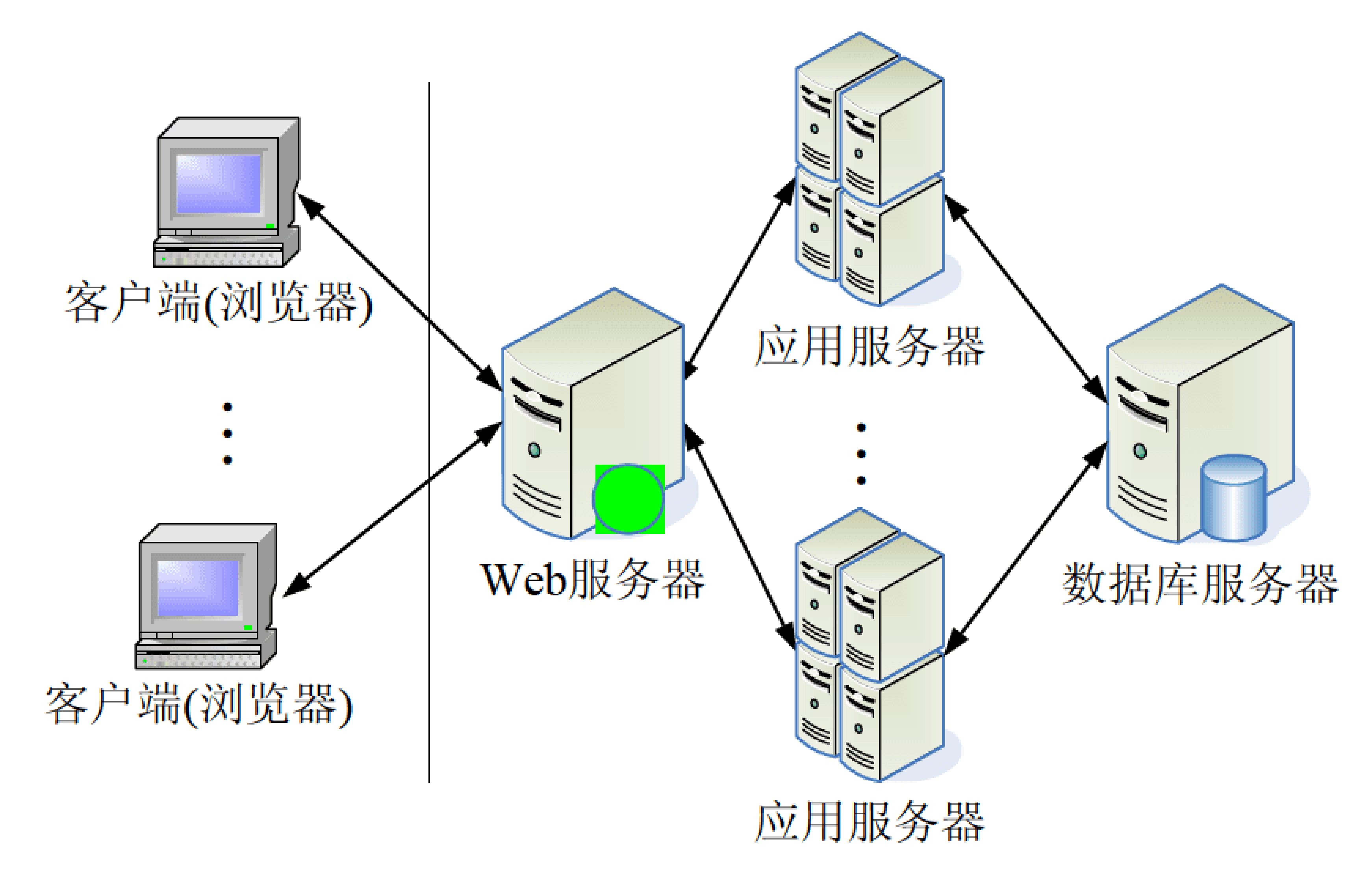


图 1-15　多层动态模式

(4) RIA(富互联网应用)

富互联网应用程序是下一代的将桌面应用程序的交互式用户体验与传统的 Web 应用的部署灵活性和成本分析结合起来的网络应用程序。富互联网应用程序中的富客户技术通过提供可承载已编译客户端应用程序(以文件形式，用 HTTP 传递)的运行环境，客户端应用程序使用异步客户/服务器架构连接现有的后端应用服务器，这是一种安全、可升级、具有良好适应性的新的面向服务模型，这种模型由采用的 Web 服务所驱动。结合了声音、视频和实时对话的综合通信技术，使富互联网应用程序(RIA)具有前所未有的网上用户体验。

RIA 具有的桌面应用程序的特点包括：在消息确认和格式编排方面提供互动用户界面；在无刷新页面之下提供快捷的界面响应时间；提供通用的用户界面特性如拖放式(drag and drop)以及在线和离线操作能力。RIA 具有的 Web 应用程序的特点包括如立即部署、跨平台、采用逐步下载来检索内容和数据以及可以充分利用被广泛采纳的互联网标准。RIA 具有通信的特点则包括实时互动的声音和图像。客户机在 RIA 中的作用不仅是展示页面，它可以在幕后与用户请求异步地进行计算、传送和检索数据、显示集成的用户界面和综合使用声音及图像，这一切都可以在不依靠客户机连接的服务器或后端的情况下进行。

可实现这类应用的技术包括 Adobe 的 Flex、微软的 SilverLight、Oracle 的 JavaFX 和 Java SWT、XUL、Bindows、Curl、Laszlo 和 MUILIB 等。

4. Web 应用框架(Web Application Framework)

这是一种开发框架技术，用来支持动态网站、网络应用程序及网络服务的开发。框架技术有助于减轻网页开发时共通性活动的工作负荷，例如许多框架提供数据库访问接口、标准样板以及会话管理等，可提升代码的可再用性。可分为基于请求的(request-based)和基于组件的(component-based)两大阵营。前者的代表有 Struts 和 Spring MVC 等，后者的成员则有 JSF、Tapestry 等。

基于请求的框架较早出现，它用于描述一个 Web 应用程序结构的概念，其和传统的静态 Internet 站点一样，是将其机制扩展到动态内容的延伸。对一个提供 HTML 和图片等静态内容的网站，网络另一端的浏览器发出以 URL 形式指定的资源的请求，Web 服务器解

读请求，检查该资源是否存在于本地，如果是则返回该静态内容，否则通知浏览器没有找到。Web 应用升级到动态内容领域后，这个模型只需要做一点修改，即 Web 服务器收到一个 URL 请求(相较于静态情况下的资源，动态情况下更接近于对一种服务的请求和调用)后，判断该请求的类型，如果是静态资源，则按上面所述处理；如果是动态内容，则通过某种机制(CGI、调用常驻内存的模块、递送给另一个进程如 Java 容器)运行该动态内容对应的程序，最后由程序给出响应，返回浏览器。在这样一个直接与 web 底层机制交流的模型中，服务器端程序要收集客户端即 Get 或 Post 方式提交的数据、转换、校验，然后以这些数据作为输入运行业务逻辑后生成动态的内容(包括 HTML、JavaScript、CSS、图片等)。

基于组件的框架采取了另一种思路，它把长久以来软件开发应用的组件思想引入到 Web 开发。服务器返回的原本文档形式的网页被视为由一个个可独立工作、重复使用的组件构成。每个组件都能接受用户的输入，负责自己的显示。上面提到的服务器端程序所做的数据收集、转换、校验的工作都被下放给各个组件。现代 Web 框架基本上都采用了模型、视图、控制器相分离的 MVC 架构，基于请求和基于组件两种类型大都会有一个控制器将用户的请求分派给负责业务逻辑的模型，运算的结果再以某个视图表现出来，所以两大分类框架的区别主要在视图部分，基于请求的框架仍然把视图也即网页看作一个文档整体，程序员要用 HTML、JavaScript 和 CSS 这些底层的代码来写“文档”，而基于组件的框架则把视图看作由积木一样的构件拼成，积木的显示不用程序员操心(当然它也是由另一些程序员开发出来的)，只要设置好它绑定的数据和调整它的属性，就可以把它从编写 HTML、JavaScript 和 CSS 界面的工作中解放出来。

5. 实际应用中的选取原则

- 首先，确定应该使用 C/S 模式还是 B/S 模式，对于某些应用场合，C/S 模式还是存在优势的。比如开发一个在 Windows 下运行的程序，或开发一个在局域网内并且只针对少量用户的程序，或者一个管理程序、后台运行程序，未必一定强求使用多层模式，因为在这种情况下 B/S 并不能带来什么突出的好处，反而会增加工作量与维护量。
- 选择了 B/S 模式进行 Web 应用开发时，要根据 Web 网站的规模、用户访问量以及要求的响应时间等几个指标来规划网站的结构模式。由于 Web 技术的发展，前面所说的静态模式现在很少采用，它已不能适应当前的用户基本要求了。
- 对于一般动态模式，要分情况对待，对于访问量很低，信息量不大，或对系统稳定性要求不是很高的情况，可以采用这种模式，因为这种模式对编程人员的素质要求不是很高，并且开发周期快，比较适用于企业内部的 Intranet 或一些访问量不大的中小网站。
- 对于一些大型的门户网站或大型的电子商务网站，由于用户访问量非常大，并且对系统的安全性以及稳定性要求都十分严格，在电子商务网站中，对数据的严谨性要求也非常严格，因此，在这几种情况下，多层动态模式就更加适合。
- RIA能实现比HTML更加健壮、反应更加灵敏和更具有令人感兴趣的可视化。RIA

允许使用一种像Web一样简单的方式来部署富客户端程序。对于那些采用C/S架构的胖客户端技术运行复杂应用系统的机构和采用基于B/S架构的瘦客户端技术部署Web应用系统的机构来说，RIA确实提供了一种廉价的选择。应用时，需要结合客户端资源和客户端的交互需求进行设计，由于可以与前面几种模式结合应用，因而可以产生多种运用模式，但通常这种应用对客户端的运算能力有一定的要求。

- 针对较为复杂的应用，可以考虑在开发过程中运用框架，而基于请求的和基于组件的方法则各有优劣。虽然一眼看上去后者有很大的吸引力，普通的 Web 开发人员只要使用专门的公司或开源组织提供的组件就可以轻松开发出好用漂亮的界面。要编写一个没有潜在问题的、跨浏览器的、显示美观、有足够灵活性并且可以调整的服务器端组件是需要高水平的技能、丰富的经验和较多时间的，即使付出这些成本，也不能完全避免使用者失望的情况。综合来看，基于请求的框架要程序员自己动手的地方比较多，但也因此可以更精细地控制 HTML、CSS 和 JavaScript 这些最终决定应用程序界面的代码，特别是如果要在界面上有创新，尝试新的视觉效果和用户操作，必然选择基于请求的框架。基于组件的框架可以提高开发界面的效率，前提是选用的组件质量优秀。
- 总而言之，技术只有与实际应用的需求紧密结合才能具有持续不断发展的生命力。针对特定应用而言，任何超前或落后的技术都将产生负面效应乃至于失败。

1.4　本章小结

Web 应用开发是目前计算机应用的热点之一。本章首先讲解了有关 Internet 和 Web 的一些基础知识；为了让读者对 Web 技术有一个较为全面的认识，分别从 Web 需求的发展、Web 应用程序发展的层面讨论了 Web 技术的本质。本章的内容为读者深入掌握 Web 技术奠定了基础。

1.5　思考和练习

1. Internet 与 WWW 有什么关系？

2. 统一资源定位符(URL)——https://www.alipay.com/aip/index.html 中，既包含了 HTTP，又包含了 WWW，三者之间是什么关系？

3. 本章提到的开发技术中哪个更好？我们应该学习哪项技术呢？

4. 你常浏览的网站属于哪一种？它们属于哪种类型？

5. 对于某一特定的网站建设需求，可以实现的技术可能有很多种，如何进行抉择？

第2章　网站策划设计与网站运行环境设置

如同在建设一栋高楼前需要做好完整的策划一样，在网站建立之前必须做好策划和较为详细的设计工作，本章给出了该设计工作的指导原则。本章还介绍了如何组织一个完整网站的开发过程及运行环境的架设，以此作为后面章节开发、运行和调试 Web 应用的基础；最后分别讨论了 Microsoft 的 IIS Web 服务器的配置、管理及安全措施。

本章要点：

- 理解网站设计的基本流程
- 网站策划的方法
- 网站设计的步骤和策略
- 网站运行环境的建立与配置
- 网站的安全与防范策略

2.1　网站设计的总体流程

一般而言，大多数网站是介绍某个特定方面内容的，具有定向性。比如国内大众所熟知的“百度”是搜索引擎站点，“阿里巴巴”是电子商务站点，“华军”是软件下载网站等。当然，除了这些专业性的站点以外，还有像“新浪”、“搜狐”、“网易”之类的门户站点，具有大而全、综合性比较高的特点。这类似于专卖店和百货商店的区别。对于大部分个人而言，通过学习和运用 Web 技术在网上建立一个展示自己的网站，这就形成了个人网站。此外，企业网站主要用于宣传某个企业的产品、形象和企业文化，游戏网站提供玩家娱乐等。总之，网站具有一定的特色，才能得到大众的认可。

一般而言，大中型网站的建设通常需要经历以下的步骤。

- 初始会商：收集待建设网站的关键信息，包括站点的目标受众，要发布的主要内容等。
- 概念开发：设计人员根据已收集到的信息，开始构思。通常，网站设计师以草图的形式呈现，其中包含整个网站的结构，不同的布局设计及导航等。
- 内容综合：当设计人员的构思得到了确认，就可以开始制作一些初始图样，之后再配合文字加以说明。

- HTML 布局和导航：若前面的设计获得了确认，则进入编制 Web 页面样例阶段，加入导航器，并进行初次的尝试和体验。
- 媒体制作：经反复修改后，站点的外观和感受最终得到了认可，此时再制作所需的各种媒体素材，并进行优化。
- 内容整合：利用各种技术将不同的媒体素材(HTML、CSS、JavaScript、JAVA、.NET、FLASH 等)，按照网站的目标有机地整合在一起。
- 网站测试：在站点被正式发布之前，测试人员要完整测试整个网站，尽量减少站点中包含的错误，并在修改后进行必要的回归测试。
- 交付：一旦测试完成，就可以正式启用该网站。这标志着网站正式进入运行阶段，当然网站的完善还需要持续做下去。

当站点启用后，还要进行持续的跟踪调查，以重新确定新的目标受众、使用该站点的方式、习惯等，根据收集到的数据开始新一轮的重新设计，如此周而复始，不断改进，总体流程如图 2-1 所示。

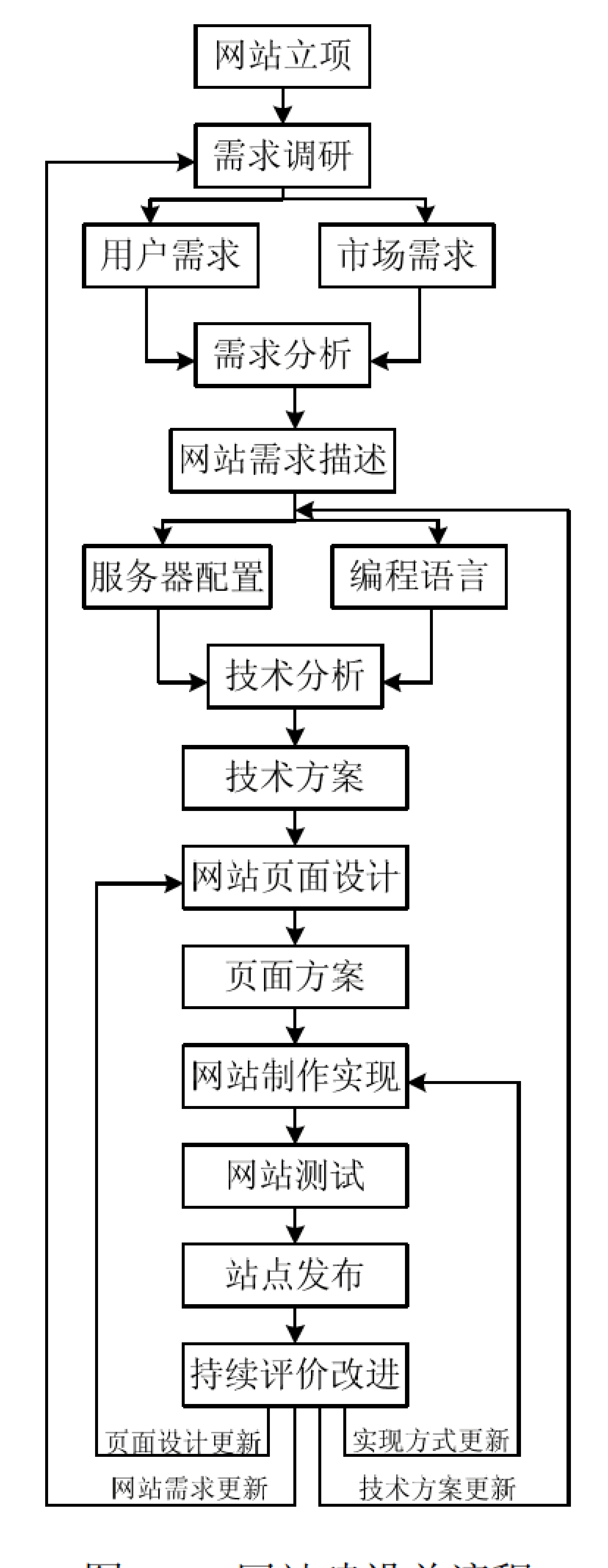


图 2-1　网站建设总流程

2.2　网站建立的前期工作——网站策划

策划是针对未来，在当前做出的决策。网站策划指的是在网站建设前根据网站建设的目的，通过市场分析，确定网站的功能，并根据需要对网站建设中的技术、内容、费用、测试、维护等做出规定。网站策划对网站的建设起到计划和指导的作用，对网站的内容和日后的改进提供方向，是成功网站平台建设成败的关键内容之一。一个网站如果想获得成功，那么建站前的策划就起着极为重要的作用，这包括必要的市场分析、明确网站建设的目的、网站的功能、规模、投入费用等。只有这样，才能避免在网站建设中的诸多问题及网站建设的失败。

网站策划活动的结束以形成完整的《网站策划书》为标志。具体而言，它包括以下几项工作。

1. 建立网站前的市场分析

- 相关行业或主题的市场是怎样的，有什么特点，该网站总体目标在互联网上开展是否恰当。
- 主要竞争对手分析、竞争对手网站情况及其网站规划、功能及效果。
- 自身条件分析，自身概况、优势，可以利用网站改善哪些条件，建设网站的能力(费用、技术、人力等)。

2. 建设网站目的及功能定位

- 为什么要建立网站？对企业而言，是为了宣传产品、进行电子商务，还是建立行业性网站？是企业形象的需要还是市场开拓的延伸？这是网站规划中的核心问题，需要非常明确和具体。网站建设的其他后续工作都是为了实现这个预期目的，不同建设目标的网站，就算其内容相似，其表达方式和实现手段也是不一样的。
- 整合现有资源，确定网站功能。根据实际的需要和时间计划，确定网站的功能：宣传、营销、服务、商务、娱乐等。
- 根据当前网站发展的阶段和目标，确定网站应起到的作用和近期建设的目标；再根据未来一段时间可能的发展状况，初步确定网站的可扩展性目标。

3. 网站的技术解决方案

在确定了网站的基本市场需求及目标后，下一步需要根据这些结论来确定网站的技术解决方案。具体来说需要做好以下几项工作。

- 决定采用自建服务器，还是租用虚拟主机。
- 确定网站的域名和名称：一个好的域名对网站的成功具有重要意义，它可以帮助记忆并突出网站形象；网站名称同域名一样重要，网站域名和名称应在网站策划阶段进行考虑。有些网站发布一段时间之后才发现网站域名或名称中存在问题，此时更改，不仅麻烦，而且让前期的网站推广工作付之东流，对网站的形象也构成一定的伤害。在早期的一些网站中这种现象较为普遍，例如：搜狐(sohu.com)曾用 sohoo.com.cn 作为域名，网易(netease.com)的前身是 nease.net，而新浪网(sina.com.cn)早期的域名是 srsnet.com。虽然目前这些网站都已发展为国内著名的门户网站，但可以看出当初他们曾经走过的弯路，这一点值得读者认真思考。
- 选择操作系统，用 UNIX、Linux 还是 Windows。逐一分析不同方案所需要的不同投入成本、功能、开发、稳定性和安全性等，并选择适合自己需求的方案。
- 采用系统性的解决方案，如 IBM、HP 等公司提供的上网方案、电子商务的解决方案？还是自己开发？
- 采取何种网站安全性措施？如何防止黑客攻击及病毒的危害？
- 选择哪种技术方案来组织开发。确定 Web 服务器的种类，如 Apache、IIS 等；选择编程语言，如 ASP、JSP、PHP、CGI、.NET 等；确定数据库产品，如 Oracle、DB2、SqlServer、Access、MySQL 等。

4. 网站内容规划

- 根据网站的目的和功能规划网站内容。如企业网站通常包括公司简介、产品介绍、服务内容、价格信息、联系方式和网上订单等基本内容。
- 是否需要提供会员注册、详细的商品服务信息、信息检索、订单确认、付款、个人信息保密措施、相关帮助等。
- 如果网站栏目比较多，则应考虑由专人负责某部分内容。

注意：

网站的内容才是网站吸引用户最重要的因素，无内容或不实用的信息只会吸引匆匆浏览的访客。需事先对受众所希望的信息进行调研，并调查访客对网站的满意度，最后根据结果调整网站内容。

5. 网页界面设计

- 网页美术设计一般要与网站整体形象一致，要符合一定的规范。注意网页色彩、图片的应用及版面规划，保持网页的整体一致性。
- 对新技术的采用要考虑网站目标、访问群体的分布地域、年龄阶层、网络速度、阅读习惯等。
- 制定网页更新计划，如规定半年或一年时间进行较大规模改版等。

6. 网站测试

网站发布前要进行细致周密的测试，以保证正常浏览和使用。测试的主要内容一般包括：

- 服务器运行稳定性、安全性。
- 程序及数据库测试。
- 各种插件、数据库、图像、链接等是否正常工作。
- 网页对不同浏览器的兼容性，以及网页在不同显示器和不同显示模式下的表现等。

7. 网站发布与推广

网站测试后进行发布的公关、广告活动，如搜索引擎登记等。网站推广活动一般发生在网站正式发布之后，当然有些网站在筹备期间就已经开始宣传。可以说，大部分的网络营销活动是为了网站推广，例如：发布新闻、搜索引擎登记、交换链接、网络广告等。因此，在网站策划阶段就应该对将来的推广有明确的认识和规划，而不是等网站建成之后才匆匆考虑。

8. 网站维护

- 服务器及软硬件的维护。对可能出现的问题进行评估，制定合适的响应时间。
- 数据库维护。有效地利用数据是网站维护的重要内容，应重视对数据库的维护。
- 内容的更新、调整等。
- 较为详细地制定网站维护的规定，将网站维护制度化、规范化。

9. 网站建设日程表

各项工作任务的开始完成时间、结束时间、负责人等，使得网站的建设能有条不紊地进行。

10. 费用明细

除了上述的技术解决方案、内容、功能、推广、测试等内容外，在上述所有过程中所涉及的财务预算也是一项重要内容，网站建设和推广在很大程度上依赖于充足的财务预算。预算应按照网站的开发周期，尽可能细致地罗列费用明细清单。

根据上述的过程，整理出最终的《网站策划书》，它应该尽可能涵盖上述的各个方面，实际上根据不同的需求和建站目的，可以灵活增减其具体内容。

注意：

这里给出的方法是针对大型网站的，一般的个人网站或小型网站实际上可以适当简化上述流程。

总之，在建设网站之初，需要进行细致的策划，这样才能在可控的时间、风险下利用一定的人手、资源按计划达到建站的目的。

2.3　网站的设计

确定了网站的总体目标及方案后，需要对实现的细节做出规定，设计过程应以用户的体验作为设计过程的出发点。网站以网络为载体，把各种信息以快捷、方便的方式传达给受众。人们对美的追求是不断深入的，网页设计同样如此，人们要求它在传达信息的同时也具有良好的视觉效果，达到形式和内容方面的统一。网页设计是网络通信技术、传播学、艺术和心理学等学科相结合的交叉学科，随着网络的日益普及而日益受到人们的重视。

网页不光是将各种信息罗列出来，能看到就行。从传播学的角度看，要考虑如何让受众更多地和更有效地接收网页上呈现的信息，给他们留下美好而深刻的印象，更好地促进网站的发展。这不仅需要从审美的方面入手，制作出清晰、美观、整体性好的页面，还需要结合使用者的心理感受，让用户得到更好的体验，减少信息交互过程中的阻力，提升网站的形象。例如，将平面设计中的节奏与韵律和骨架的组织形式融入到网页呈现中，使内容繁多的页面更有条理，浏览时主次分明。当然这种美首先建立在页面的内容充实且实用的基础上，一个内容空洞无物的网页即使做得再漂亮也是不会吸引人的。从这个意义上来看，内容是网站的生命线，其他方面是让网站更受欢迎的催化剂。

而网页的从无到有，从满足基本的功能需要到追求更高层次的需要，这是一个循序渐进的过程。这使人不由得想起工业革命前夕，很多现代的产品那时候都没有，没有现成的模式可以参照，产品的设计都是从满足基本的功能需要出发，所以做出来的产品比较粗糙，冷冰冰，毫无生气；但经过商业竞争和工业化大生产，在不断改进产品功能的同时，大幅

度改进了产品的外观，使产品更符合审美的需要，使用起来更方便，从而造就了今天琳琅满目且美观实用的各种产品。其实，网站也是如此，在满足了基本的功能性需求之后，为了突出自己的特色，突出自己的优势，必须从审美上入手。

注意：

设计网页并不是一个十分复杂的过程，但想要设计出合理而精美的网站，则需要经过严谨的理性分析、敏锐的观察，以及感性的审美和创意。

设计过程需要首先理解用户的习惯，主要包括以下几个方面。

(1) 不同用户的阅读方式

首先，读者是随意的和被动的。网站需要面对不同类型的读者，且人们常常受到各种干扰。通常离线阅读者更加专注于内容，而在线读者往往关注某项具体任务，缺乏耐心。

(2) 用户的阅读习惯

Poynter Institute 新闻学院曾进行了一项研究，发现大约 50%的用户不会以逐个单词的方式来阅读 Web 文本，而是来回扫视。

(3) Web 内容的非线性

与印刷介质不同的是，Web 站点通过超链接将所有内容组织为非线性的结构。人们往往通过搜索引擎或者其他站点的链接进入网站中的某一页。因此对于导航和相关链接的组织，就需要做出全局的设计。

(4) 屏幕阅读方式

绝大多数的 Web 用户采用的是直接读取屏幕内容的阅读方式，这对于眼睛而言通常较为劳累。因此网页需要在视觉上设计为尽可能平和的方式。文本需要分解为更小的单元，以利于扫视。

(5) 考虑视觉障碍用户的使用

如果网站可能针对包括视觉障碍的用户，则需要考虑通过听觉和触觉来表现网站内容，以便这部分用户使用。

(6) 交互措施

与其他媒体不同的交互方式，使得 Web 非常容易开展用户互动工作，这些意见和建议对于网站的长远发展具有非常重要的意义。

网站的设计需要从不同角度分别进行，以下从网站设计的不同层面来介绍网站设计的基本方法。

2.3.1　网站的 CI 形象设计

所谓的 CI(Corporate Identity)，借用了广告的术语，它是通过视觉来统一企业的形象。现实生活中的 CI 策划比比皆是，杰出的例子如：可口可乐公司，具有全球统一的标志，色彩和产品包装，给我们的印象极为深刻，类似的例子还有 SONY、三菱和麦当劳等。

一个杰出的网站，和实体公司一样，也需要整体的形象包装和设计。准确、有创意的 CI 设计，对网站的宣传推广能起到事半功倍的效果。在网站主题和名称定下来之后，需要

思考的就是网站的 CI 形象，以下是一些具体做法和步骤。

1. 设计网站的标志(logo)

首先需要设计制作一个网站的标志(logo)。如同商标一样，它是站点特色和内涵的集中体现，看见 logo 就能让大众联想到这个站点。注意：此处的 logo 不是指 88×31 的小图标，而是网站的标志。

标志可以是中文、英文字母、符号、图案，也可以是动物或人物等。例如：soim 是用英文字符 soim 作为标志的，新浪用字母 sina+眼睛作为标志。标志的设计创意来自网站的名称和内容。常用的设计思路有以下几种。

- 最常用和最简单的方式是：用网站的英文名称做标志。采用不同的字体、字母的变形、字母的组合等。
- 网站有代表性的人物、动物、花草，可以用它们作为设计的蓝本，在此基础上加以卡通化和艺术化，例如：迪士尼的米老鼠，搜狐的卡通狐狸等。
- 专业性的网站，可以采用本专业有代表性的物品作为标志。比如：中国银行的铜板标志、奔驰汽车的方向盘标志。

注意：

很多人对网站标志的形状存在误区，认为网站标志不能做成竖长方形的，而必须为横长方形的；网站标志的位置必须在页面的左上角等；在专家眼里，这些想法太教条了，页面的设计是可以具有个性化的。

2. 设计网站的主色调

网站给人的第一印象来自视觉冲击，确定网站的主色调是相当重要的一步。不同的色彩搭配产生不同的效果，并可能影响到访问者的情绪。

“标准色彩”是指能体现网站形象和延伸内涵的色彩。举个实际的例子：IBM 的深蓝色、肯德基的红色条型、Windows 视窗标志上的红蓝黄绿色块，都使我们觉得很贴切，很和谐。如果将 IBM 改用绿色或金黄色，会产生什么感觉？

一般来说，一个网站的标准色彩不超过 3 种，太多则让人眼花缭乱。标准色彩要用于网站的标志、标题、主菜单和主色块，给人以整体统一的感觉。至于其他色彩也可以使用，只是作为点缀和衬托，绝不能喧宾夺主。一般来讲，适合于网页标准色的颜色有蓝色，黄/橙色，黑/灰/白色三大系列色。

3. 设计网站的标准字体

和标准色彩一样，标准字体是指用于网站的标志、标题、主菜单的特有字体。一般而言，网页默认的字体是宋体。为了体现网站“与众不同”和其特有的风格，可根据需要选择一些特殊的字体。例如，为体现专业性可使用粗仿宋体，体现设计精美可以用广告体，体现亲切随意则可以用手写体等。当然，完全可以根据自己网站所表达的内涵，选择更贴切的字体。目前常见的中文字体有二三十种，英文字体有近百种，网络上还有许多专用英

文艺术字体下载，要寻找一款满意的字体并不算困难。

注意：

使用非默认字体通常只能采用图片的形式，因为用户的计算机中往往没有安装这种特别的字体，且大多数用户不会为了浏览个别网站而安装这些特别的字体，这样就会导致意想不到的显示效果。

4. 设计网站的宣传标语

宣传标语也可以说是网站的精神、网站的目标。用一句话甚至一个词来高度概括，类似实际生活中的广告金句，如雀巢的“味道好极了”；麦斯威尔的“好东西和好朋友一起分享”；Java 的“一次编写，到处运行”等。

以上的标志、色彩、字体和标语，是一个网站树立 CI 形象的关键，确切的说是网站的表面文章，设计并完成这几步，网站将脱胎换骨，整体形象有一个提高。这个过程可形象地类比为：由一个实在而土气的农民转变为一位西装革履的职业人士。

2.3.2　网站的总体结构设计

网站需由多名设计人员协同工作，最后进行合成。如果毫无规范和约束，任由设计者按照自己的设想进行设计，就容易落入结构上混乱、维护上困难的境地。因此，在网站的设计过程中统一和规范开发人员的设计行为是非常必要的。

1. 网站的目录结构

网站的目录结构是指网站所有文档在站内目录的组织和存放结构。大型网站的目录数量多、层次深、关系复杂，必须严格按照一定的规则来存放，而网站的目录结构又是一个容易被忽视的问题，许多网站设计者未经周密规划，随意创建目录，给日后的维护工作带来不便。目录结构的好坏，对用户来说并没有什么太大的影响，但对于站点本身的维护，如上传、内容的扩充和移植等有着重要的影响。因此，必须合理定义目录结构并组织好所有文档。以下是一些在设计工作中被证明是行之有效的做法。

- 不要将所有文件都存放在根目录下。一些网站设计人员为了方便，将所有文件都放在根目录下。这样造成了文件管理混乱，当项目开发到一定阶段后，设计者不能分辨哪些文件需要编辑和更新，哪些无用的文件可以删除，哪些是相关联的文件，影响工作效率。此外也常常造成上传速度变慢：服务器通常会在根目录下建立一个文件索引。如果将所有文件都放在根目录下，造成单个目录下包含大量文件，那么即使只上传更新一个文件，服务器也需要将所有文件再检索一遍，建立新的索引。文件数量越大，则等待的时间就越长。此处切实可行的做法是尽可能减少根目录中文件的数目。
- 按栏目内容建立子目录。建立子目录的惯例是按主菜单的栏目来建立，例如：网页教程类站点可以根据技术类别，分别建立相应的子目录，像 Flash、DHTML 和 JavaScript 等；而企业站点就可以按公司简介、产品介绍、价格、在线订单、意见

反馈等栏目建立相应的目录。对于其他的次要栏目，如新闻、行业动态等，由于内容较多，需要经常更新，可以建立独立的子目录。而一些相关性强，不需要经常更新的栏目，例如，关于本站、关于站长、站点成长经历等则可以合并放在某个统一的目录下。所有的程序一般都存放在特定目录下，以便于维护和管理。例如，CGI 程序放在 cgi-bin 目录下，ASP 网页放在 asp 目录下。所有供客户下载的内容应该放在一个目录下，以方便系统设置文件目录的访问权限。

- 在每个主目录下都建立独立的 images 目录来存放相应的图片。在默认的设置中，每个站点根目录下都有一个 images 目录，可以将所有图片都存放在这个目录里。但是，这样做也有不方便的时候，当需要将某个主栏目打包供用户下载，或者将某个栏目删除时，图片的管理相当麻烦。经过实践发现，为每个主栏目建立一个独立的 images 目录是最方便管理的。而根目录下的 images 目录只是用来放首页和一些次要栏目的图片。
- 目录的层次不要太深或太扁平。为了使维护和管理方便，目录的层次建议不超过 3 层，宽度最好不超过 15 个。
- 目录和文件不要使用中文来命名。使用中文目录可能对网址的正确显示造成困难，特别是某些 Web 服务器不支持中文目录和文件。
- 不要使用过长的目录，尽管服务器支持长文件名，但是太长的目录名既不便于记忆，也不便于管理。尽量使用意义明确的目录，如将 Flash、DHTML、JavaScript 作为名称来建立目录，以便于记忆和管理。

随着网页技术的不断发展，利用数据库或者其他后台程序自动生成网页的用法越来越普遍，而网站的目录结构的设计也必将上升到一个新的层次。

2. 合理设计网页间的逻辑结构

有研究表明：当用户碰到陌生且内容较为繁杂的信息时，用户会不自觉地建立一个模型。这个模型表征用户对这些信息的理解，用户用此方式来梳理这些复杂的信息，该模型可以评估哪些信息是新的，并据此给予重点关注。因此，一个成功的网站结构设计应尽量与大多数目标受众的预期相符合；在内容上组织良好的逻辑结构让用户能准确、快速地找到自己所要的信息。图 2-2 至图 2-4 分别表示了 3 种不同的逻辑结构。其中值得提倡的是图 2-4 的结构，它具有合理的深度和宽度分布，构造了比较均衡的结构。

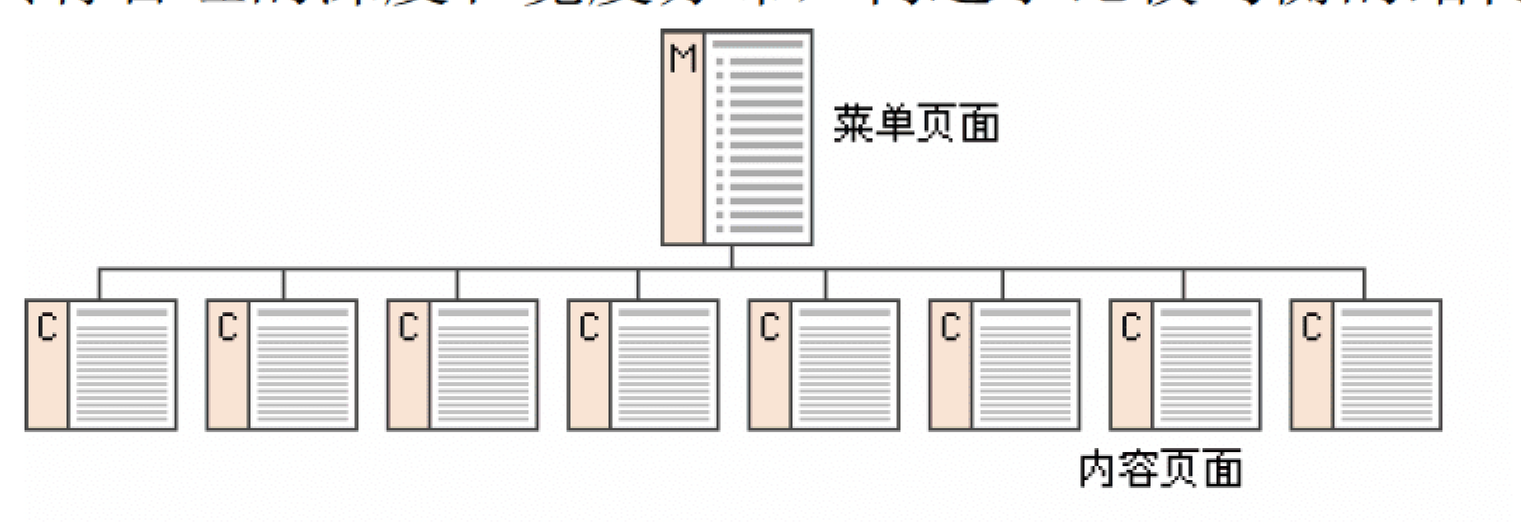


图 2-2　过于扁平的逻辑设计

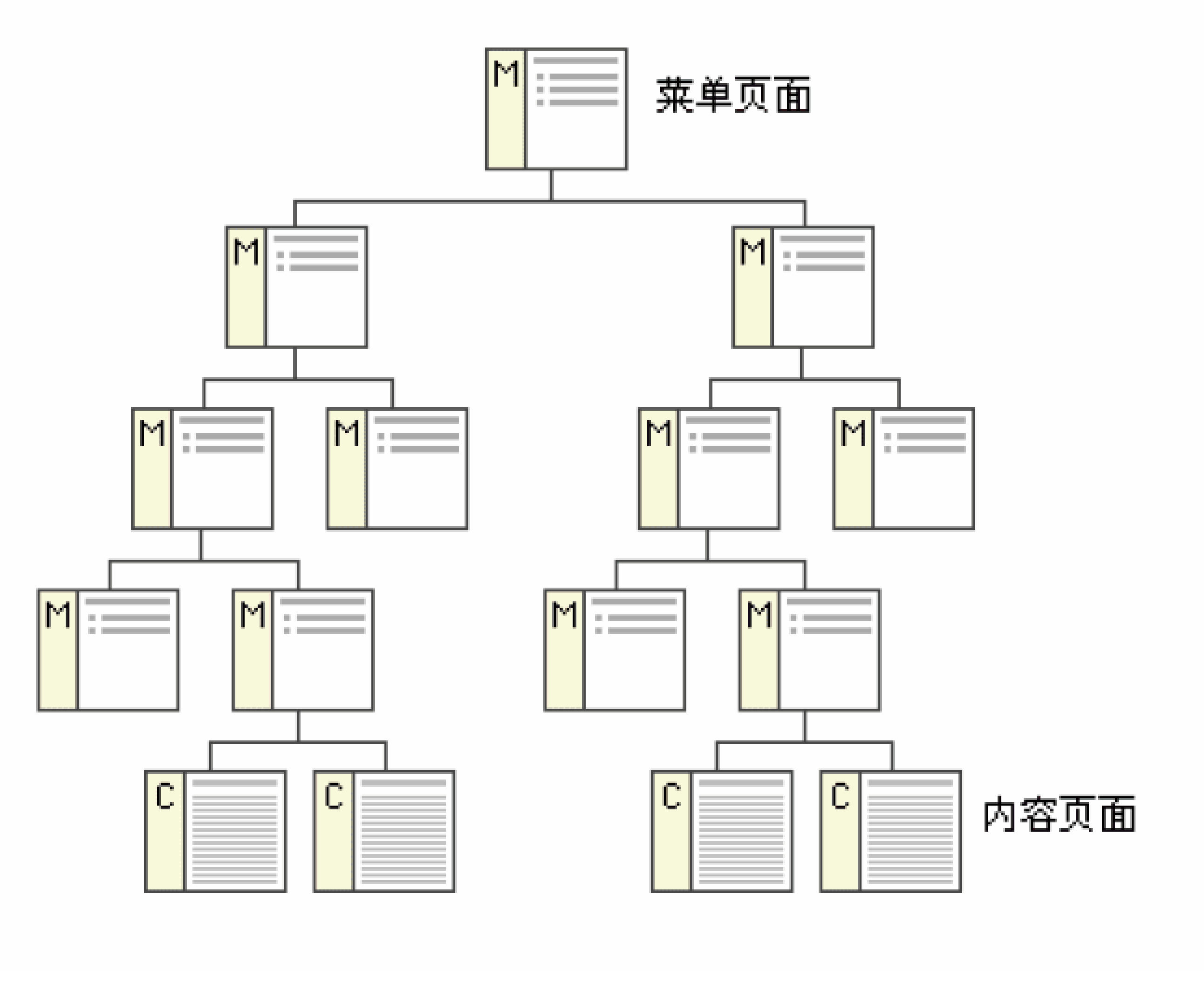




图 2-3　过于深的逻辑设计

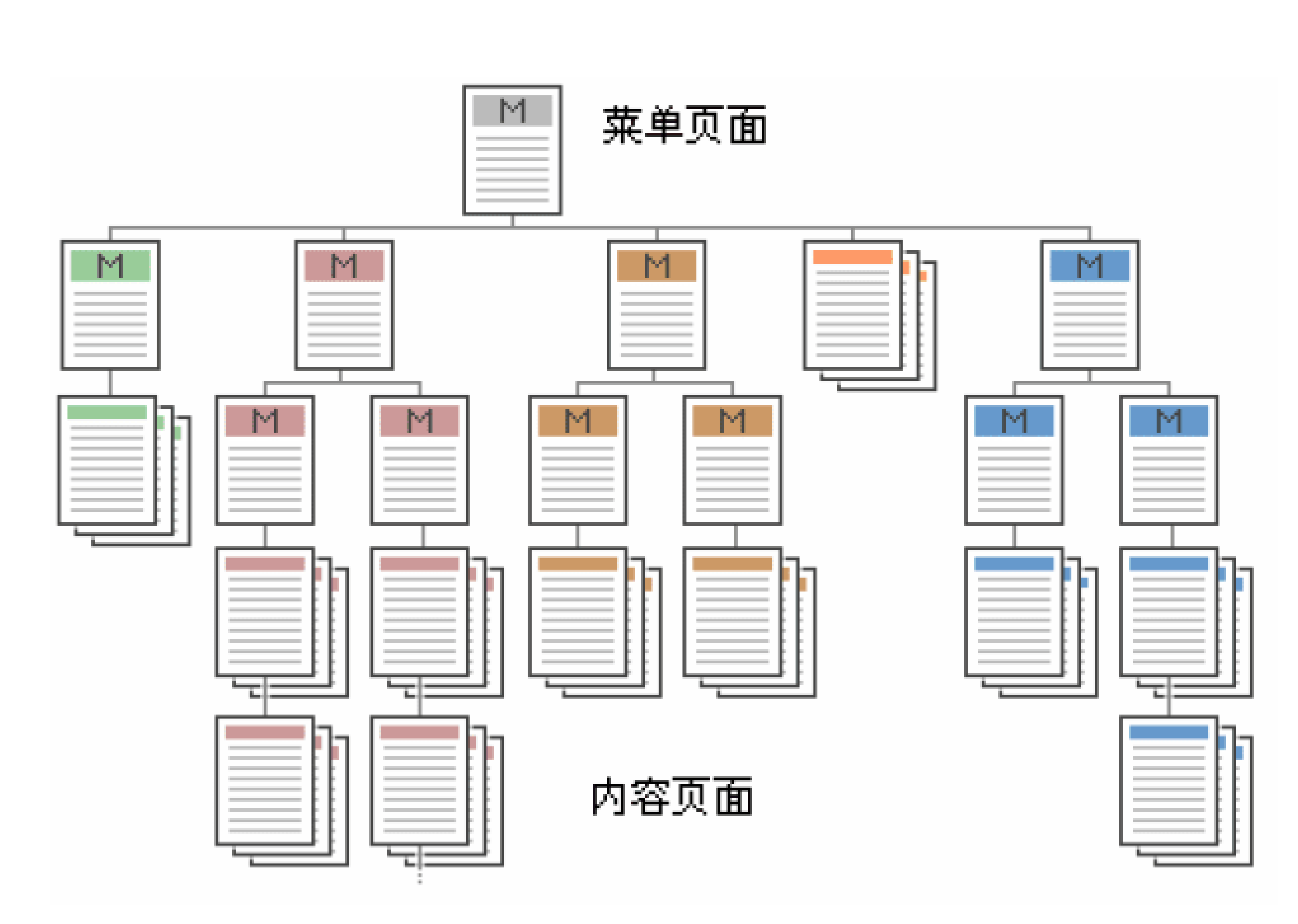


图 2-4　均衡的逻辑设计

注意：

网站的总体结构的设计不是孤立的，实际上它是与其他方面的设计，如导航、版面、色彩等方面结合起来的，而网站的内容和目标才是进行设计的依据。

2.3.3　网站的版面设计

版面指的是浏览器看到的完整的一个页面(可以包含框架和层)。因为每台计算机显示器分辨率不同，所以同一个页面在用户浏览时可能出现 1366×768 像素、1024×768 和 1920×1280 像素等不同尺寸。

1. 布局

布局是以最适合浏览的方式将图片和文字排放在页面的不同位置。这里的“最适合”是一个不确定的形容词，谁也不能给出一个绝对正确的答案，且每个人对于相同的版面布局也会有不同的看法。版面布局是一个创意的问题，但要比站点整体的创意容易、有规律得多。

如同传统的报纸杂志编辑一样，也可以将网页看作一张报纸、一本杂志来进行排版布局。虽然动态网页技术的发展使设计人员开始趋向于高级的思维方式，但是固定的网页版面设计基础仍然是必须学习和掌握的。网页的排版与书籍杂志的排版又有很多差异。印刷品都有固定的规格尺寸，网页则不然，它的尺寸是由读者来控制的，这使网页设计者不能精确控制页面上每个元素的尺寸和位置。而且，网页的组织结构不像印刷品那样为线性组合，这给网页的版式设计带来了一定的难度。总之，它们的基本原理是共通的，在设计网页时还需要领会其中的要点并举一反三。

有研究表明，中国用户更倾向以扫描、打圈的方式阅读网页。同时，中国用户更喜欢往返于各个内容区域，说明他们阅读时比较随意。相反地，美国用户比较关注细节，也很少这看一眼那看一眼。对于采用整体思维方式的中国用户来说，他们一般是以跳跃性的浏览方式，更倾向于整体阅读，针对该类用户，内容安排可以更为灵活。而对于美国用户来

说，他们主要采用分析性思维方式，在页面的布局上要做到非常清晰，尤其导航非常重要，每个部分都要有各自的特点。了解人脑(视觉、听觉)如何接收信息，其目的是使设计适应人的自然特性，满足用户的要求。

对互联网产品用户界面的设计，既要满足用户感官认知特征，又要满足用户心理认知特征，这样的界面才是以用户为中心的界面，也是设计者所追求的界面。对于版面布局的总体要求建议如下：

- 加强视觉效果；
- 加强文案的可视度和可读性；
- 具有统一感的视觉；
- 新鲜和个性是布局的最高境界。

2. 布局设计的步骤

为了能做好这项工作，在此首先了解版面布局的一般步骤。

- 草案：新建页面就像一张白纸，没有任何表格、框架和约定俗成的东西，可以尽可能地发挥创作者的想象力，将 “景象”画上去(在此建议使用纸和笔，用其他软件亦可)。在创造阶段可以不讲究细腻和工整，不必过多考虑细节功能，只用简单的线条勾画出创意的轮廓即可。尽可能多画几种，再从中选定一个满意的作为继续创作的脚本。
- 粗略布局：在草案的基础上，将确定需要放置的功能模块安排到页面上。尤其需要注意的是，功能模块主要包含网站标志、主菜单、新闻、搜索、友情链接、广告条、邮件列表、计数器、版权信息等。

注意：

这里必须遵循突出重点、平衡协调的原则，将网站标志、主菜单等最重要的模块放在最显眼、最突出的位置，然后再考虑次要模块的排放。

- 定案：将粗略布局精细化，具体化。这一步骤需要凭借智慧和经验，旁敲侧击多方联想，才能创作出具有创意的布局。

在布局过程中，需要遵循的原则还有以下几项。

- 正常平衡：亦称“匀称”。多指左右、上下对照形式，主要强调秩序，能达到安定诚实、信赖的效果。
- 异常平衡：即非对照形式，突破一般的平衡和韵律，此布局能起到强调性、不安性和吸引眼球的效果。
- 对比：指不仅利用色彩、色调等技巧来表现，在内容上也可涉及古与今、新与旧和贫与富等的对比。
- 凝视：利用页面中人物视线，使用户仿照跟随的心理，以达到注视页面的效果，一般多用明星凝视。
- 空白：空白有两种作用，一方面对比于其他网站以表示突出和卓越，另一方面也

表示了网页的品位及优越感，这种表现方法对体现网页的格调十分有效。

- 尽量用图片解说：对不能用语言叙述或语言无法表达的情感，图片特别有效。图片解说的内容，可以传达给用户更多的心理因素，有时能带来震撼的效果。

以下的几条设计原则属于设计的细节，虽然枯燥，但是如果能领会并活用到页面布局里，也能起到画龙点睛的效果。

- 如果网页的白色背景太虚，则可以加些色块。
- 如果版面零散，可以用线条和符号串联。
- 如果左面文字过多，右面则可以插一张图片保持平衡。
- 如果表格太规矩，可以试试改用导角。

3. 版面布局形式

经常被用到的版面布局形式如下。

- “T”结构布局：就是指页面顶部为横条网站标志+广告条，下方左面为主菜单，右面显示内容的布局，因为菜单条背景较深，整体效果类似英文字母“T”，所以称之为“T”形布局，如图 2-5 所示。这是网页设计中用得最广泛的一种布局方式。这种布局的优点是页面结构清晰，主次分明，是初学者最容易上手的布局方法。其缺点是规矩呆板，如果细节色彩上不注意，很容易让人“看之无味”。



图 2-5　“T”形结构布局

- “口”形布局：这是一个形象的说法，就是页面一般上下各有一个广告条，左面是主菜单，右面放友情链接等，中间是主要内容。这种布局的优点是充分利用版面，信息量大。缺点是页面拥挤，不够灵活。也有将四边空出，只用中间的窗口

型设计，如图 2-6 所示的网页。



图 2-6　“口”形型布局

● “三”形布局：这种布局多用于国外站点，国内用得不多。特点是页面上横向两条色块，将页面整体分割为四部分，色块中大多放广告条，如图 2-7 所示。

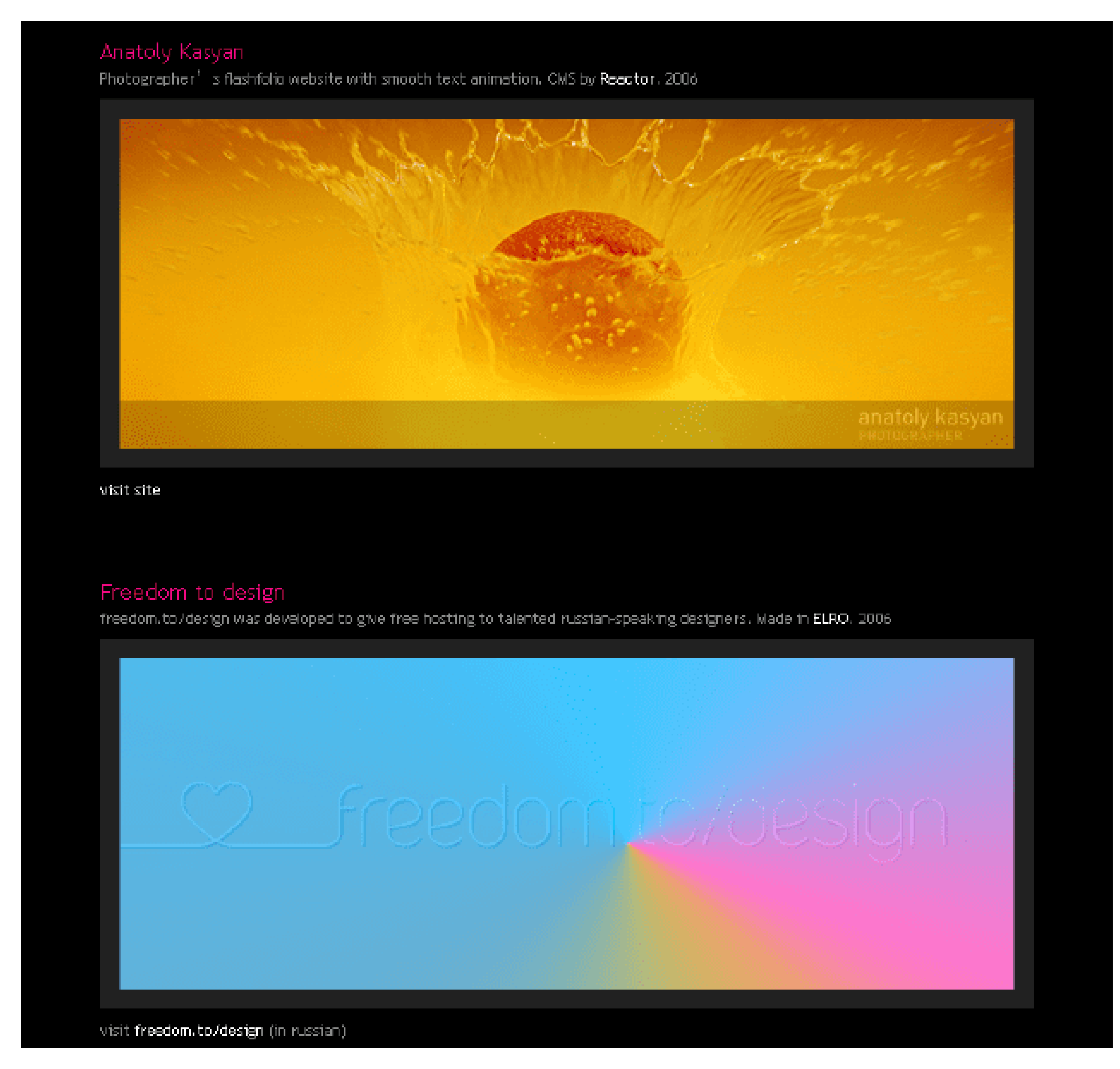


图 2-7　“三”形布局

- 对称对比布局：顾名思义，采取左右或者上下对称的布局，一半深色，一半浅色，一般用于设计型站点，如图 2-8 所示。优点是视觉冲击力强，自由活泼，风格独特，在有限的空间内可显示较多文字和图像；缺点是将两部分有机地结合比较困难。

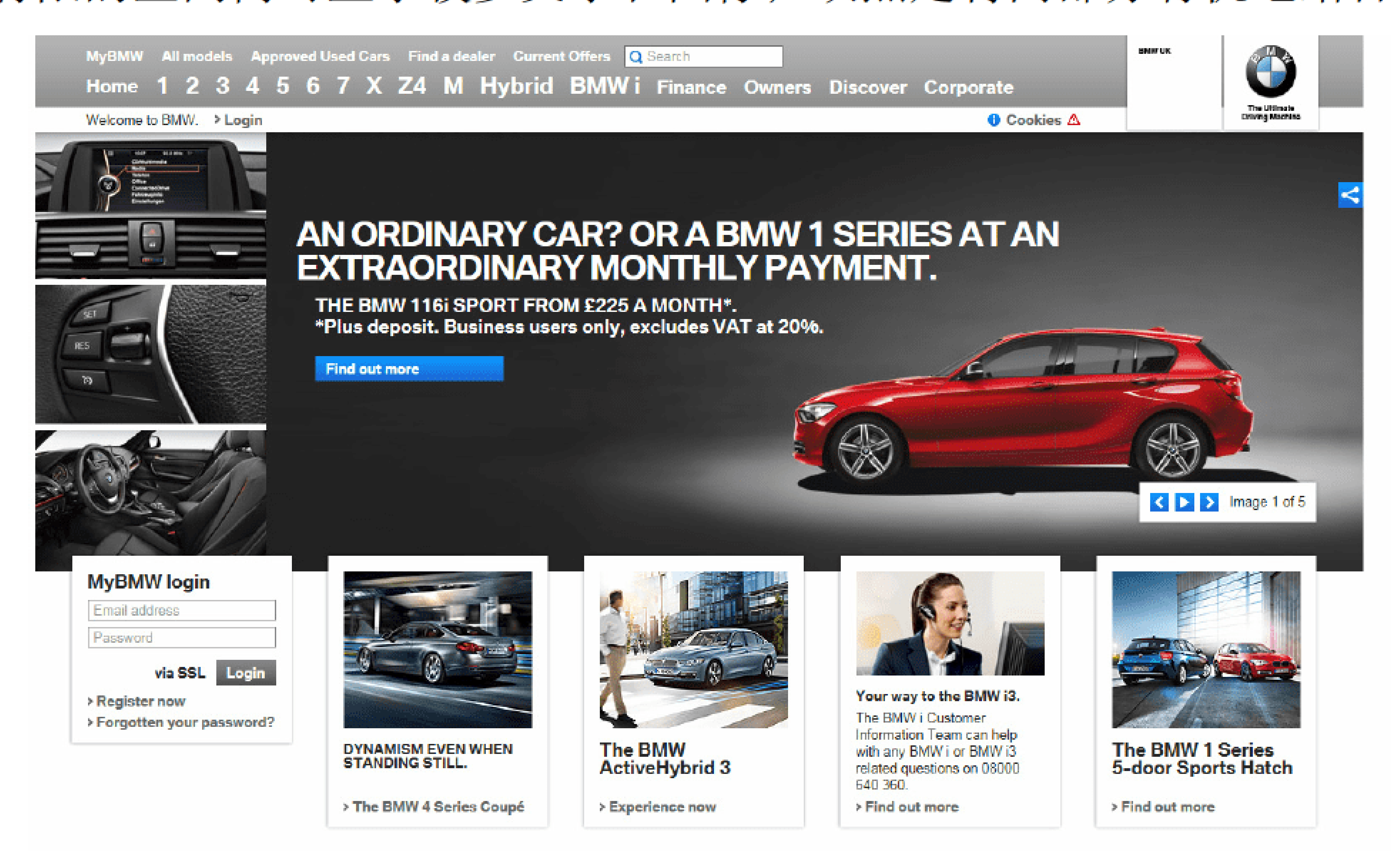


图 2-8　对称对比布局

- POP 布局：POP 引自广告术语，就是指页面布局类似一张宣传海报，以精美的图片作为页面设计的中心。使用多列在同一空间呈现更多内容，更多的网站在朝着多列布局、流动布局，以及适应多种终端的响应性设计方向发展。这种设计可以让网页的信息呈现更加美观易读，更具可用性，界面设计上将更加灵活与规范。常用于时尚类等站点，如图 2-9 所示。其优点显而易见，如漂亮吸引人；缺点就是速度慢，但作为版面布局还是值得借鉴的。

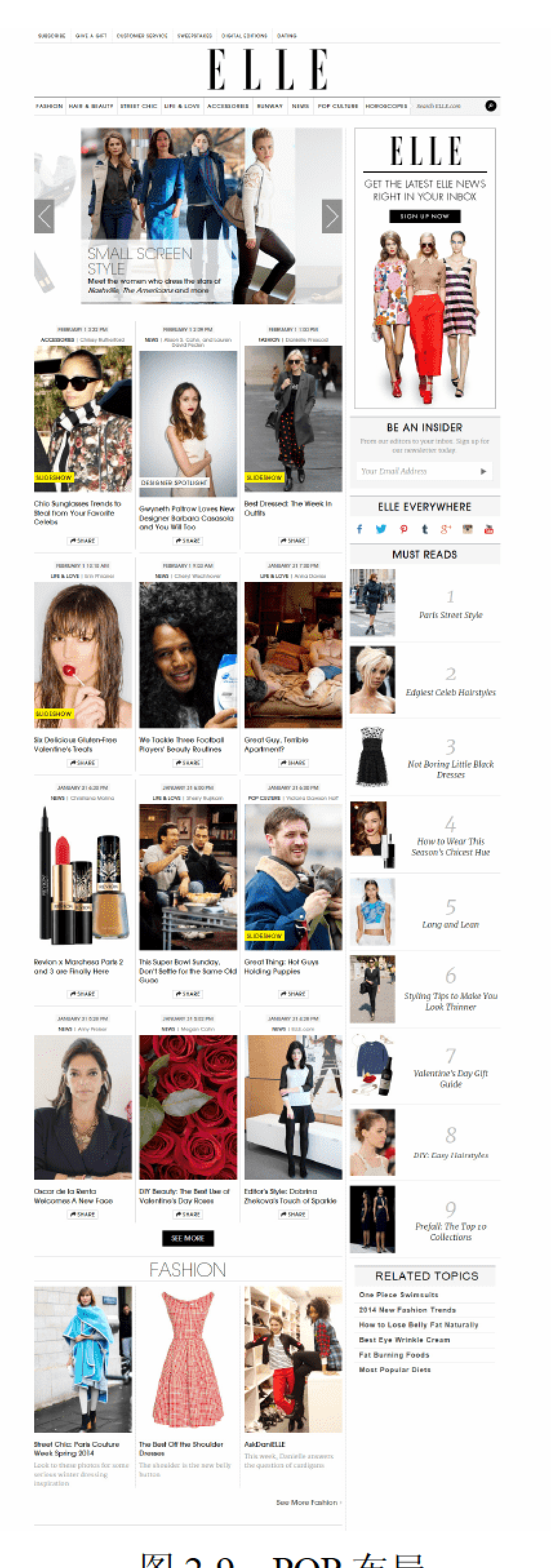


图 2-9　POP 布局

- “同”字形布局：“同”字结构名副其实，采用这种结构的网页，往往将导航区置于页面顶端，一些如广告条、友情链接、搜索引擎、注册按钮、登录面板、栏目条等内容置于页面两侧，中间为主体内容。这种结构比左右对称结构要复杂一点，不但有条理，而且直观，有视觉上的平衡感。但是这种结构也比较僵化，如图 2-10 所示。该布局优点是一目了然，结构清晰、对称、主次分明，目前得到广泛的应用。缺点是太过于规矩、沉闷，没有个性，不能够很好地激发用户的兴

趣，需要善于运用色彩的变化细节来调剂。



图 2-10　“同”字形布局

- “回”字形布局：“回”字形实际上是“同”字形布局的一种变形，即在“同”字形结构的下面增加了一个横向通栏，这种变形将“同”字形结构不是很重视的页脚利用起来，这样增大了主体内容，合理地使用了页面有限的面积，但是这样往往使页面充斥着各种内容，显得拥挤不堪，如图 2-11 所示。



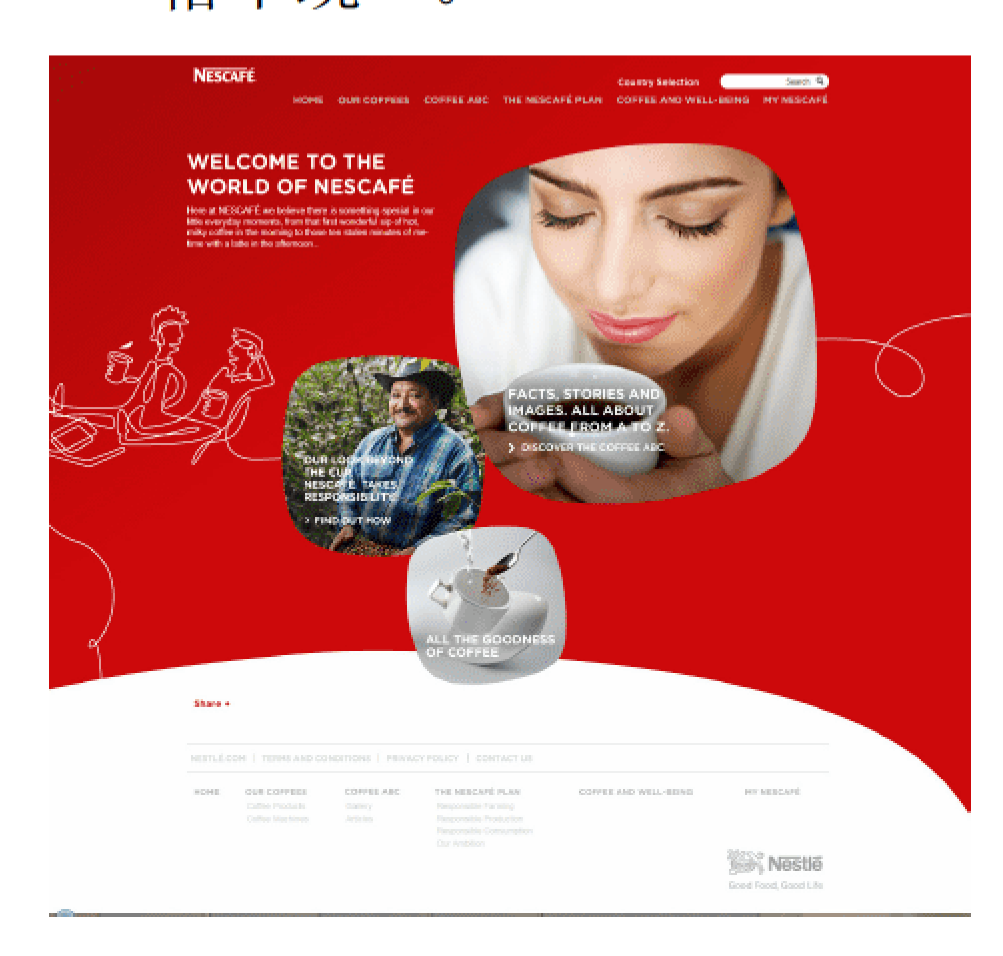
图 2-11　“回”字形布局

- “匡”字形布局：与“回”字形布局一样，“匡”字形结构其实也是“同”字形结构的一种变形，也可以认为是将“回”字形结构的右侧栏目条去掉得出的新结构，这种结构是“同”字形结构和“回”字形结构的一种折中，这种结构承载的信息量与“同”字形相同，而且改善了“回”字形的封闭形结构，如图 2-12 所示。



图 2-12　“匡”字形布局

- 其他布局：更多的网站并没有一定的规律，它们只是在排版布局中加入自己的各种想法，形成了各具特色的风格，这种结构的随意性特别大，颠覆了从前以图文为主的表现形式，将图像、Flash 动画或者视频作为主体内容，其他的文字说明及栏目条均被分布到不显眼的位子，起装饰作用，这种结构在时尚类网站中使用的非常多，尤其是在时装、化妆用品的网站中。这种结构富于美感，可以吸引大量的用户欣赏，但是却因为文字过少，而难以让用户长时间驻足，另外起指引作用的导航条不明显，而不便于操作，如图 2-13 所示。采用这种结构时，这里该放什么，这里不该放什么，必须做到事先胸有成竹，否则可能变得混乱，使得整个网站风格不统一。



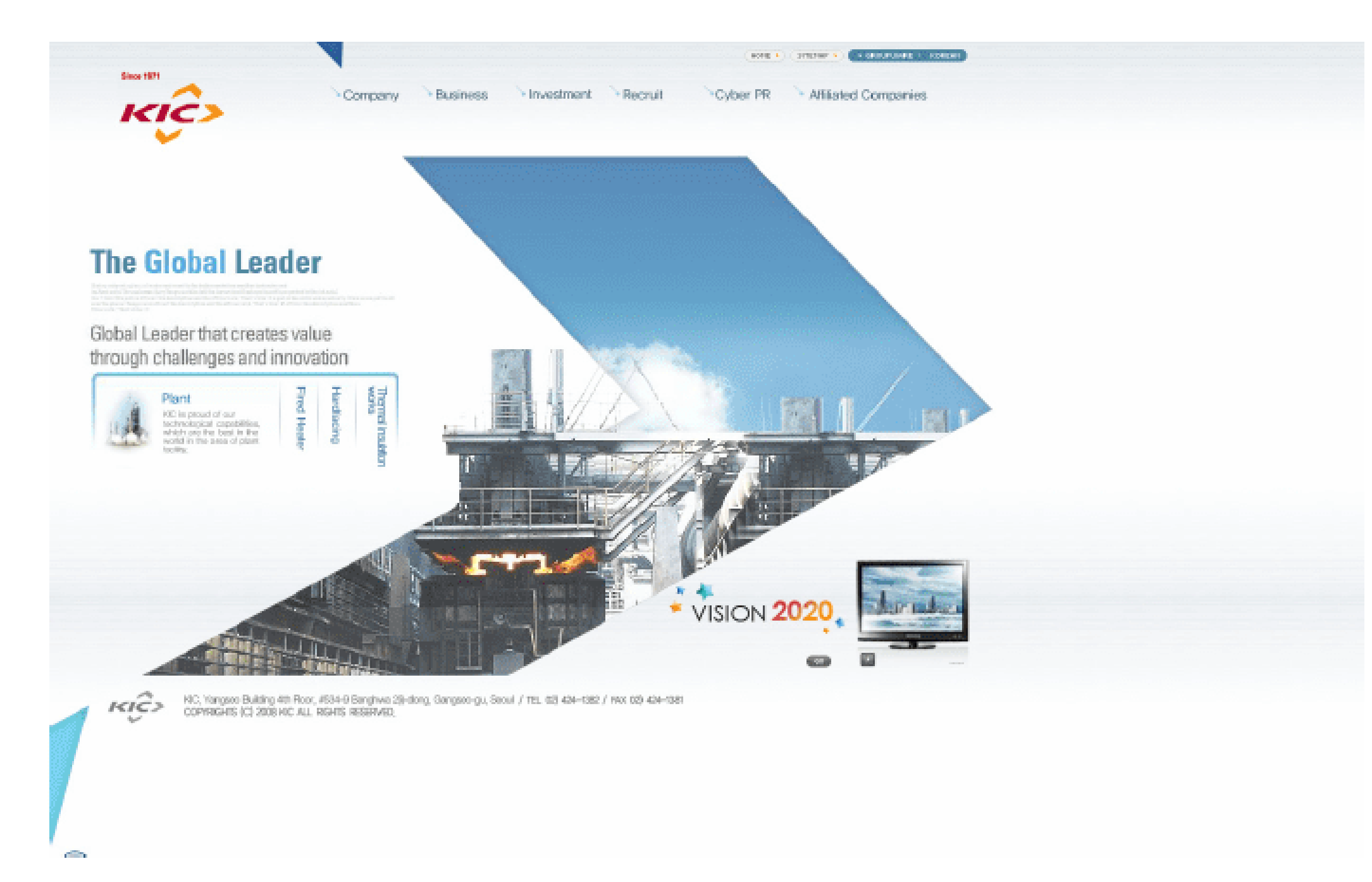


图 2-13　其他布局

以上总结了网页设计中常见的布局，其实还有许多别具一格的布局，关键在于针对内容的创意和设计。

2.3.4　网页的色彩设计

1. 216 种安全色彩

不同的平台(Mac、PC 等)有不同的调色板，不同的浏览器也有自己的调色板。这就意味着对于一幅图，在 Mac 上的浏览器中显示的图像有可能与它在 PC 浏览器中显示的效果差别很大。选择特定的颜色时，浏览器会尽量使用本身所用的调色板中最接近的颜色。如果浏览器中没有所选的颜色，就会通过抖动或者混合自身的颜色来尝试重新产生该颜色。

为了解决 Web 调色板的问题，人们一致通过了一组在所有浏览器中都类似的 Web 安

全颜色。这些颜色使用了一种颜色模型，在该模型中，可以用相应的十六进制值 00、33、66、99、CC 和 FF 来表达三原色(RGB)中的每一种。这种基本的 Web 调色板将作为所有的 Web 浏览器和平台的标准，它包括了这些十六进制值的组合结果。这就意味着，我们潜在的输出结果包括 6 种红色调、6 种绿色调、6 种蓝色调。6×6×6 的结果就给出了 216 种特定的颜色，这些颜色就可以安全地应用于所有的 Web 中，而不需要担心颜色在不同应用程序之间的变化。

2. 色彩的意义

所选择的颜色要适合目标受众、能表达客户希望网站建立者传达的信息，能符合用户在网站所获得的整体感受的期望。暖色能带来阳光明媚的情绪，用在希望带来幸福快乐感觉的网站上是明智的。例如，在 2009 年全球经济不太好的时候，黄色变成了网页设计中非常流行的色彩，因为公司希望顾客在他们网站上有阳光和舒适的感受。冷色最好是用在想要表达出专业或整洁感觉的网站上，以呈现出一个冷静的企业形象。冷色表达出权威、明确和信任的感觉。例如，冷静的蓝色用在许多银行的网站上，比如交通银行等。冷色运用在以乐观为主题的网站上是不明智的，因为用户会得到错误的印象。

设计师在决定了一个网站风格的同时，也决定了网站的情感，而情感的表达很大程度上取决于颜色的选择。颜色是很有力的工具，几种常用的颜色所表达的含义分别如下。

- 红色：是一种激奋的色彩。刺激效果，能使人产生冲动、愤怒、热情和活力的感觉。它象征着火和力量，还与激情和重要性联系在一起，还有助于激发能量和提起兴趣。红色的负面内涵是愤怒、危急和生气，紧急情况下，还表示愤怒，这也源于红色本身的热情和进取。
- 绿色：介于冷暖两种色彩之间，显得和睦、宁静、健康和安全。它和金黄、淡白搭配，可以产生优雅、舒适的气氛。绿色象征着自然，并且有一种治愈性的特质。它可以用来象征成长与和谐。绿色让人感到安全，因此医院经常使用绿色。另一方面，绿色的是金钱的象征，表达着贪婪或嫉妒。它也可以被用来象征缺乏经验或初学者需要成长(“没有经验的绿色”)。
- 橙色：这是一个欢快的色彩，具有轻快、欢欣、热烈、温馨和时尚的效果，象征着幸福、快乐和阳光，能唤起孩子般的生机。它虽然没有红色那么积极，但是它也有一部分这样的特质，刺激着人们的心理活动。有时候它也象征着愚昧和欺骗。
- 黄色：黄色是一种幸福的颜色，它的明度最高，代表着积极、喜悦、智慧、光明、能量、乐观和幸福。而一个昏暗的黄色则带来负面的感受：警告、批评、懒惰和嫉妒。
- 蓝色：是最具凉爽、清新和专业的色彩，它和白色混合，能体现柔顺、淡雅、浪漫的气氛，也象征着信任和可靠性。因此，这是一个和平、平静的颜色，散发着稳定和专业性，因此它普遍运用于企业网站。一个冷调的阴影能带来蓝色消极的一面，象征着抑郁、冷漠和被动。
- 白色：白色不是色轮的一部分，象征洁白、明快、纯真、清洁和天真，它还传达

着干净和安全。相反，白色还可以被认为是寒冷和遥远的象征，代表着冬天的严酷和痛苦的特质。

- 黑色：虽然黑色不是色轮的一部分，但它仍然可以被用来暗示感觉和意义。它往往是与权力、优雅、精致和深度联系在一起。据说在面试时穿黑色服装可以表现出应聘者是一个有力量的个体，网站也是同样的道理。黑色具有深沉、神秘、寂静、悲哀和压抑的感受，也常被看作是负面的，因为它与死亡、神秘和未知联系在一起，这是悲伤、悼念和悲哀的颜色，因此在运用时必须明智选择。
- 灰色：具有中庸、平凡、温和、谦让、中立和高雅的感觉。

有调查表明：随着网页制作经验的积累，设计者用色呈现了这样的趋势：单色→五彩缤纷→标准色→单色。一开始因为技术和知识缺乏，只能制作出简单的网页，色彩单一；在有一定基础和材料后，希望制作一个漂亮的网页，将自己收集的最好的图片，最满意的色彩堆砌在页面上；但是时间一长，却发现色彩杂乱，没有个性和风格；第三次重新定位网站时，选择好切合自己的色彩，推出的站点往往比较成功；当最后设计理念和技术达到顶峰时，则又返璞归真，用单一色彩甚至非彩色就可以设计出简洁精美的站点。

3. 色彩搭配

色彩搭配的基本原理如下。

- 色彩的鲜明性：鲜艳的网页色彩容易引人注目，吸引用户的注意力。
- 色彩的独特性：与众不同的色彩，可加强对网站的印象。
- 色彩的合适性：色彩和所表达的内容相适合。如用粉色体现女性站点的柔性。
- 色彩的联想性：不同色彩会产生不同的联想，蓝色想到天空，黑色想到黑夜，红色想到喜事等，选择色彩要和网页的内在主题相适应。
- 如果用一种色彩，即先选定某一种色彩，然后调整其透明度或者饱和度(通俗的说就是将色彩变淡或者加深)，产生新的色彩再用于网页。这样的页面看起来色彩统一，有层次感。
- 如果打算用两种色彩，则先选定第一种色彩，然后再选择它的对比色作为第二种颜色。这样做可以使整个页面色彩丰富且不花哨。
- 即使打算采用多种颜色，也不要将所有颜色都用到，尽量控制在 3 种色彩以内，否则会很乱。
- 尽量使用一个色系。即使用相关的色彩，如淡蓝、淡黄和淡绿；或者土黄、土灰和土蓝。
- 背景和前文的对比尽量要大(绝不要采用花纹繁复的图案作背景)，以便突出主要文字内容及重点。

注意：

配色就是需要处理好色彩的统一与变化、秩序与多样性的关系。色彩只有在与周围的环境相搭配的情况下才能产生美感。

4. 不同色彩在网页设计中的应用

- 确定网页的主色调，选择衬托色彩，凸显网站内容和主题。一般网站都是以浅色调为主的居多，如人人网以较浅的蓝色为基调，豆瓣网以浅绿色为基调，沪江网以浅灰色为基调等。以浅色为底，会给人柔和素淡的感觉，配合深色的文字，让人视觉上得到舒适的感觉。浅色的基调有利于整体页面的搭配，便于突出网页的重点内容，方便阅读。其他的如背景图片等次要内容，不能喧宾夺主，要采用不那么抢眼的颜色。而需要得到突出表现的内容，就适宜采用明亮的色彩，或者与背景色对比度大的色彩，对网页的用户产生强烈的视觉冲击，但不宜过多使用，否则反而会让页面变得繁杂混乱。
- 在网页中运用同色系的色彩。同种色彩搭配是指首先选定一种色彩，然后调整其透明度和饱和度，将色彩变淡或加深，而产生新的色彩，这样的页面看起来色彩统一，具有层次感。邻近色是指在色环上相邻的颜色，如绿色和蓝色、红色和黄色即互为邻近色。采用邻近色搭配可以使网页避免色彩杂乱，易于达到页面和谐统一的效果。使用同色系的色彩，会让人觉得页面赏心悦目，留下好印象。同色系的色彩易于搭配，不会使得页面显得杂乱，也不会让人视觉疲惫，可以大面积使用。但同色系的色彩搭配也存在弊端，它容易使得网页色彩单调，令人觉得乏味，为了解决这个问题，网页设计师可以采取邻近色或者局部使用对比色的图片来增加页面整体变化，带来一种轻松活泼的感觉。
- 页面中对比或互补色的运用。一般来说，色彩的三原色(红、黄、蓝)最能体现色彩间的差异。色彩的强烈对比具有视觉诱惑力，可以突出重点，产生强烈的视觉效果。通过合理使用对比色，能够使网站特色鲜明、重点突出。通过使用对比互补的色彩，能够塑造轻松活泼，以及运动感强的网页效果，它适宜体现轻松主题的网站，既色彩丰富，又协调悦目。虽然对比色带来了如此的好处，它同样也存在着缺陷。过于丰富的色彩会造成视觉的疲惫和重点不清晰，譬如，过于丰富的背景色彩会影响前景图片和文字的取色，严重时会使文字融于背景中不易辨识。所以，在设计网页时，设计者只用一到两种色彩占据主导地位，其他的对比色作为陪衬或点缀来调节整体效果，大面积的色彩填充宜用低比度色彩，所以背景色用单一的色彩为佳。

总而言之，色彩的运用方式多种多样，目的是要让网页设计达到实用和审美的统一与和谐，要解决设计中的色彩运用的具体问题，就需要在长期的实践中不断积累经验和学习前人总结的解决方法。

2.3.5　网站导航设计

网站导航就是帮助人们找到他们在网页浏览时的路标，它是网站设计不可缺少的基础元素之一。导航不仅仅是对整个网站信息结构的分类和组合，也是浏览网站的路标。进入网站后，人们通常会寻找导航条，并由此直观地了解网站的主要内容和信息分类的方式。

事实上，许多用户都是以一种跳跃的方式来访问网站的内容。为了使用户不在网站中迷失方向，最好的办法是为网站设计科学有效的导航系统，以下是设计导航的一些基本原则。

(1) 将它放置在重要的位置上。导航是页面中重要的视觉元素，因此应该将它放置在明显、易找、易读的区域，以便用户在进入网站的第一时间就可以看到。

(2) 注意超链接颜色与一般文字的区分。WWW 语言——HTML 允许网页设计者区分一般文字与超链接的颜色，以便突出和区分不同网页元素的功能。

(3) 测试所有的超链接与导航按钮的有效性。导航在增加了用户使用网站的便利的同时，也带来了发生错误的可能。网站发布之后，第一件该做的事，是测试每一页的每一个超链接与每一个导航按钮的有效性。彻底检验有没有失败的导航和无法连接到该链接的网页，避免出现“File Not Found”的错误信息。

注意：

也不是每个网站都必须采用导航。如果网站没有那么多的超链接项，不妨采用列表的方式，将它们清楚地列在某个选单页或目录页上，这样既不妨碍内容的顺畅，又呈现一目了然的导航。还可以采用列表的方式，该内容在 3.3.4 节超级链接中进行了介绍，请读者参阅第 3 章“HTTP 协议及其开发与 HTML 语言基础”。

(4) 让超链接的字串长短适中。抓住能传达主要信息的字眼为超链接的锚点(anchor)，可有效地控制超链接字串的长度，避免过长(如整行、整句都是锚点字串)或过短(如仅一个字当作锚点)，而妨碍用户阅读或点取。

(5) 对较长的文本提供必要的链接。将篇幅过长的文件分隔成数篇较小的网页可大大增加界面的亲和力，但在导航按钮与超链接的配置上，网页设计者则要更细心周全地安排，使得读者不论身处网站的哪个层次，依然能够快速便捷地通往其他任何一个页面。对此，网页设计者应特别注意以下情况。

- 提供“上一页”、“下一页”、“回子目录页”与“回首页”的导航按钮或超链接。在一系列具有前后关系的顺序文件里，各网页都至少应提供“上一页”、“下一页”、“回子目录页”与“回首页”的导航按钮或超链接，这样可使读者能够立即得知自己所在的页面，是属于一份较大文件内的一小部分(考虑、体贴读者不是从主页顺序链接至此页，而是依循别的网站的某个链接跳跃链接至此)。并且可以借由这些链接随时参考连接“上一页”、“下一页”与本页的连贯内容；直接点取“回子目录页”查寻其他相关的标题，或直接跳跃至主页，浏览其他不同项目的信息。
- 简明扼要地标明此页、上一页与下一页文件的标题或内容梗概。在一系列具有前后连续顺序的文件里，各网页都应加上一个具有说明性的标题，使读者一目了然，马上抓住这一页的重点。而完善的导航系统除了提供“上一页”、“下一页”的导航按钮或超链接外，还应该添加简洁的上一页与下一页标题、内容提要等，使读者在尚未浏览这些网页时，也能先大概地了解自己将链接到一个什么样的网页。
- 提醒读者某一系列文件已到尽头。当读者已达某一系列文件的最后一页时，网页

设计者应提供一小段告示提醒读者，同时不再提供“下一页”的导航按钮或超链接。但基于网页界面设计的一致性，或许有些网页设计者并不希望在同一系列的最后一篇网页里忽然少了一个先前每页都有的“下一页”导览按钮(尤其是精心设计过的图形化导航按钮)。为达此目的，可以将最后一页的“下一页”导航按钮的颜色变暗些，且该超链接不可单击，并提供一小段文字来提醒读者，该文件已到尽头，不再有“下一页”的内容。

- 明确表示出用户当前所在的位置。由于一些设计师的疏忽，往往会将“所在位置”忘记，而使用户不知道他们自己在站点中的位置。虽然 URL 提供了一个精确的位置，但大多数用户不能理解。一个高级的网页标签形式加入了关于位置的许多信息，它直接显示了用户在站点中的位置，其一般形式为“首页>作品展示>网站设计>作品 1”。

(6) 在较长的网页内提供目录与标题。理想的网页长度一般不超过三四个屏幕。但是有时网页必须要做得很长，那么此时可以在此网页最上方提供目录，在相关内容处标上大小标题，以便阅读。尤其重要的是，按照设计惯例可以在这些标题和目录上设置锚点，在目录处可以直接链接到这些锚点。

(7) 暂时不提供到尚未完成网页的超链接。超链接或导航按钮应能引导读者到正确的目标，那种事先描述得很精彩的链接，单击后除了“正在建设中”外看不到任何内容的体验，是任何用户所不希望的。如果急欲发布站点，但仍有少数几个网页尚未完成，建议不生成链接，等完成后再开放。

(8) 不要在一篇短文里提供太多的超链接。适当、有效率地使用超链接，是一个优良的导航系统不可或缺的条件之一。但滥用超链接会造成短短的一篇文章里处处是链接，损害了网页的流畅与可读性。一般而言，文章里提供的文字超链接最好不超过 10 个。连续地出现两三个文字式的超链接，很容易被误认为只是一个长度较长的超链接，读者很容易忽略掉，这样的导航便失去了意义。

注意：

有效的导航是让内容有效凝聚的方式，导航设计体现的是网站内容的分类。为了便于建设和管理，通常可以和网站的物理结构——目录相一致。

2.3.6　网站信息的可用性设计

一个以信息为主要内容的网站，页面中的信息组织形式、版式和分类等直接关系到用户的浏览体验。目前常用的信息呈现方式有以下几种。

(1) 文字列表形式。此形式在网站中使用率最高，优点是可以在“寸土寸金”的有限空间内尽可能地放更多内容。缺点是文字列表的简单重复方式比较单调，重点不突出，阅读比较困难，浏览体验大大下降，尤其在中文的显示方式下这些缺点则更加突出。

目前的解决办法一般是在每一行文字前加一个修饰点，用于引导用户的浏览；或者加分割线将每一行都分割开来，控制行间距等，如图 2-14 所示。

当然，在某一个区域内进行这样的补偿设计是很有效果的。但是，如果整个页面大量地采用这种方式，则还是会影响用户的阅读。

(2) 图片形式。经研究发现，网站上的图片被关注的程度高于文字信息。图片传达给用户的是感官的直接刺激，用户不需要动脑筋就可以通晓，用户在大脑里可以迅速地提取相关的信息，采用图片形式来展现信息的网页，如图 2-15 所示。

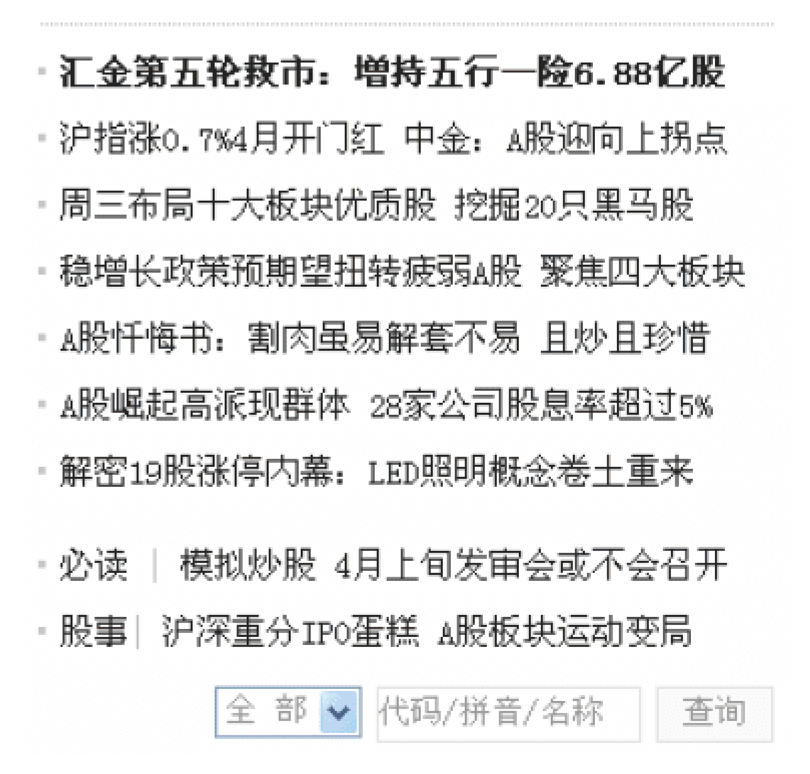


图 2-14　文字列表形式



图 2-15　图片形式

用图片显示不失为一个好的方式，但是图片占用的空间大，网络传输速度较慢，而且部分图片令人费解的问题也还是存在的。

(3) 图片加文字内容形式。单从用户体验的角度上讲，这种形式算是最佳的浏览方式。图片在用户的脑中形成的是具象的信息，文字、语言在用户的脑中形成的则是抽象的信息。而这种形式对用户在“行”和“意”上都做了考虑。也就是说，用户看图片和文字时，大脑的工作区域是不同的，最终会达到一个“图文并茂”的效果，如图 2-16 所示。

这种方式的优点是能提供用户良好的浏览体验，让用户不用抽象地思考，大大减少了用户思考的时间，提高了网站的可用性。缺点是这样做需要太大的空间(一般一条文字加图片的信息可以放 10 条左右的纯文字信息)，导致无法放入更多的信息资源，不能在一个页面大量地使用，一般用于需要突出的重要信息。

(4) 迷你块。上面的几种方式都存在一定的缺陷，那到底有没有一种两全其美的方法呢？答案是肯定的，这就是“迷你块”。它是把一条信息的图片和文字以列表的形式展现出来，每一条信息都由一张很小的图片和一条文字标题组成，达到了图文并茂的效果。再利用其本身所占区域较小的特点，组成一个列表，以上所说的问题就都迎刃而解了，如图 2-17 下部区域所示。



图 2-16　图片加文字内容形式　　图 2-17　迷你块

不过，迷你块这种折中的做法，其效果也是折中的：单位空间内不会比只有文字列表丰富；图片的大小限制，不会特别清晰明了等。

总之，如何让用户在获取网站信息的时候减少思考的负担，是广大界面设计者的设计宗旨。而方法的选取也视用户群体、网站类型等各方面因素所决定。

2.4　网站的建立——IIS 的安装与配置

Web 服务器是通过软件来实现的，常用的软件包括 IIS 和 Apache。下面主要讨论 IIS 软件的安装和配置。如果用户使用非 Windows 的操作系统来建设网站，请参考相关的说明文档和书籍进行配置。通常 IIS 只在 Windows 平台上运行，而其他常用 Web 服务器可能会提供不同平台的安装包。

2.4.1　IIS 的安装

IIS(Internet Information Server，即：互联网信息服务)是一种 Web(网页)服务组件，其中包括 Web 服务器、FTP 服务器、NNTP 服务器和 SMTP 服务器，分别用于网页浏览、文件传输、新闻服务和邮件发送等方面，它使得在网络(包括互联网和局域网)上发布信息成了一件很容易的事。目前运行 IIS 比较理想的平台是 Windows Server 2008。Windows Server 2008 服务器中集成了 IIS 7.0 版。但是在安装上述部分操作系统时，除了专门用于网站服务的 Web 版外，IIS 可能是不被默认安装的。因此对于其他版本，用户必须要手动安装 IIS，只有当计算机上安装了 IIS 之后，这台计算机才能成为一台 Web 服务器。

注意：

由于 IIS 属于 Windows 操作系统附带的软件，在 Windows XP 安装光盘中也附带有 IIS 软件，但在 XP 中只能支持 10 个并发用户，其他管理功能方面也受到了不少限制。

在 Windows Server 中，安装 IIS 有 3 种途径：利用“管理您的服务器”向导，利用控制面板下“添加或删除程序”的“添加/删除 Windows 组件”功能，或者执行无人值守安装。以控制面板下“添加或删除程序”为例，其操作步骤如下。

(1) 打开 Windows“控制面板”中的“添加和删除程序”，然后选择“添加/删除 Windows 组件”，此时会弹出一个“Windows 组件向导”对话框，如图 2-18 所示。

(2) 在此对话框中单击“详细信息”按钮，打开“应用程序服务器”对话框，可以对该组件进行详细配置，如图 2-19 所示。



图 2-18　“Windows 组件向导”对话框

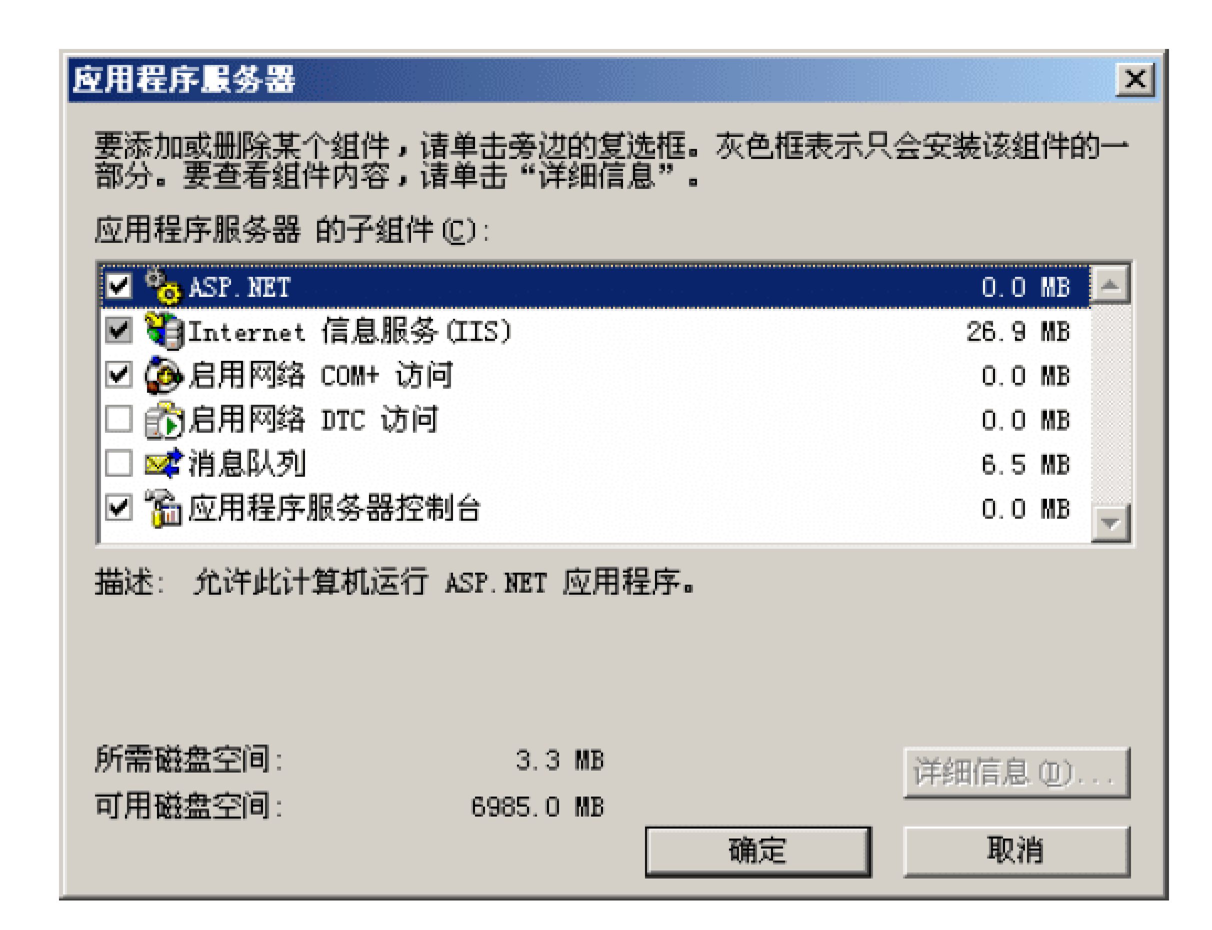


图 2-19　“应用程序服务器”对话框

(3) 单击“确定”按钮，再在图 2-18 的对话框中单击“下一步”按钮即可完成安装，如果安装成功，可以看到如图 2-20 所示的对话框。

上述的安装过程正常结束后，系统会自动在系统盘新建网站目录，默认的发布目录为 C:\Inetpub\wwwroot。

注意：

如果用户使用的是其他版本的 Windows 系统，也可以先在控制面板中检查该系统是否已经安装了 IIS。

鉴于目前仍有不少用户在使用 Windows 7 或 Windows 8 等类型的操作系统，下面以 Windows 7 系统为例简单说明在这类系统上安装 IIS 的方法。首先在“控制面板”中单击“程序与功能”图标，在弹出的窗口中单击左侧的“打开或关闭 Windows 功能”，此时会看到类似如图 2-21 所示的组件显示。



图 2-20　IIS 安装顺利完成



图 2-21　Windows 7 操作系统下的 IIS 安装

单击 Internet 信息服务前面的复选框，可以看到一个灰色背景的选中状态，这表示默认安装不是全部安装。如果想变更所安装的功能，可以单击单选框前面的“+”号，这样可以看到如图 2-22 所示的安装选项，其中有支持 ASP 及.NET 开发的选项。单击其中的“确定”按钮后，等待安装过程完毕即可实现所选功能的安装。单击其中的“确定”按钮后，等待安装过程完毕即可实现所选功能的安装。

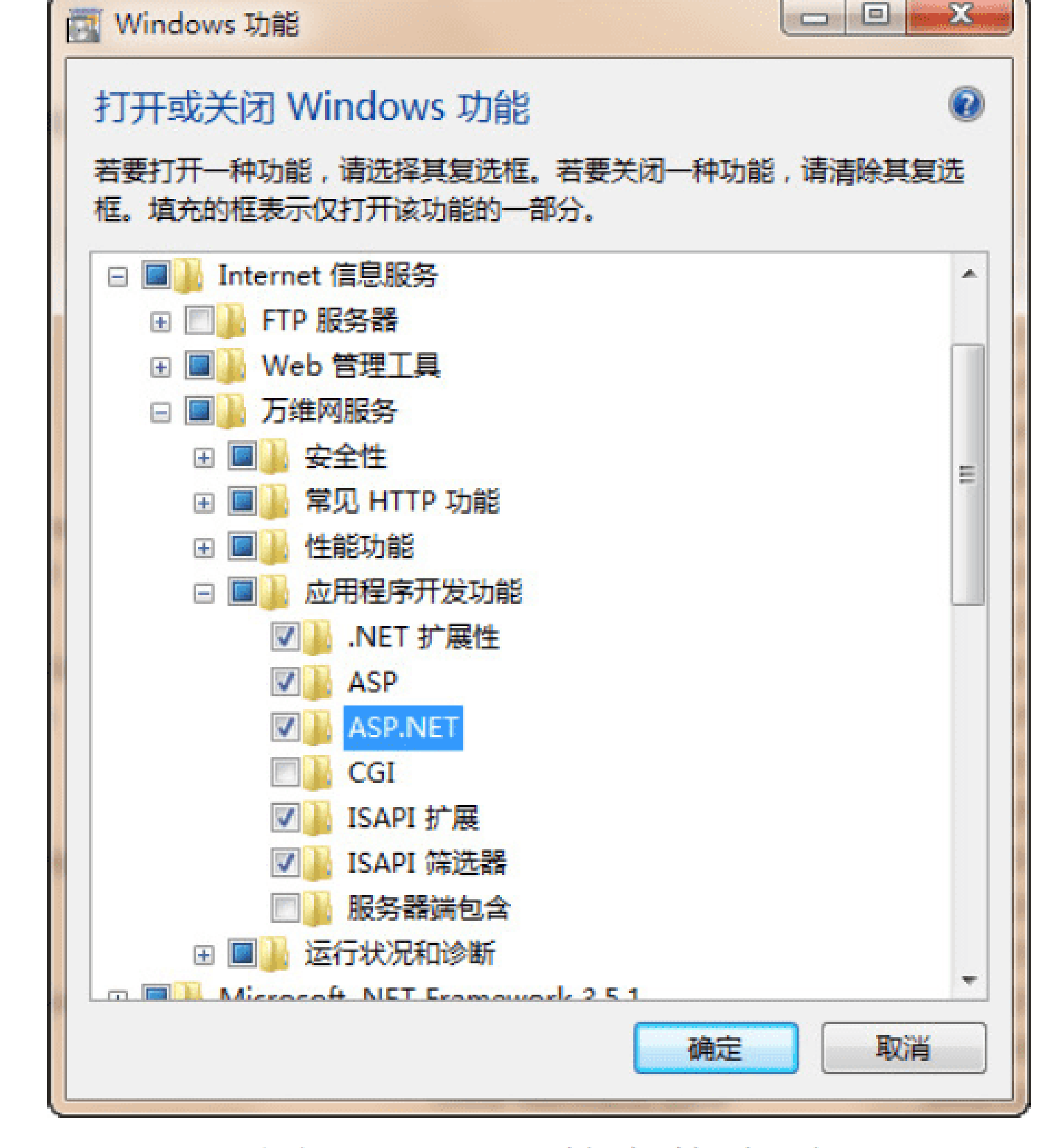


图 2-22　IIS 的安装选项

2.4.2　使用 IIS 建立站点

IIS 安装完成后，在系统的“管理工具”下会增加“Internet 信息服务(IIS)管理器”的选项，它用于监视、配置和控制 Internet 信

息服务、创建 Web 站点、FTP 站点等，首先需要打开并连接到欲管理的目标服务器。

一旦连接成功，就可以进行相应的建立站点和配置等操作，下面就介绍如何利用 IIS 在本机上创建 Web 站点，同时介绍关于站点的基本配置。

(1) 打开“开始”菜单→“管理工具”下的“Internet 信息服务(IIS)管理器”(有些版本的 Windows 中可以在“控制面板”中找到“管理工具”)，可以对 IIS 服务进行管理。打开“Internet 信息服务(IIS)管理器”后出现的控制台主窗口如图 2-23 所示。

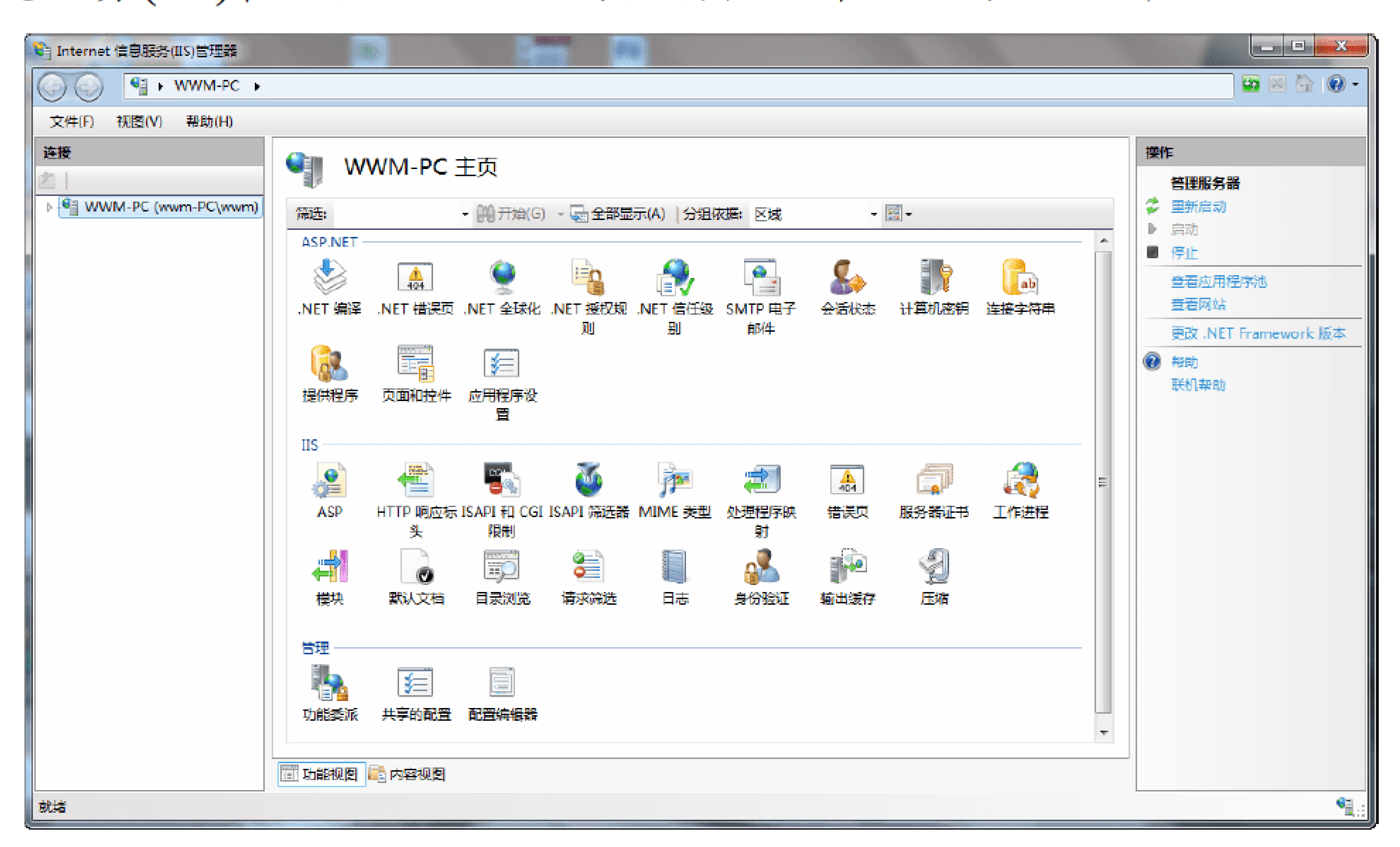


图 2-23　“Internet 信息服务(IIS)管理器”控制台主窗口

(2) 在图 2-23 所示的控制台左侧列表中的本机名称或弹出的“网站”上右击，在弹出的弹出快捷中选择“添加网站”选项，此时弹出“添加网站”对话框，在其中填写必要信息后即可创建新网站，如图 2-24 所示。

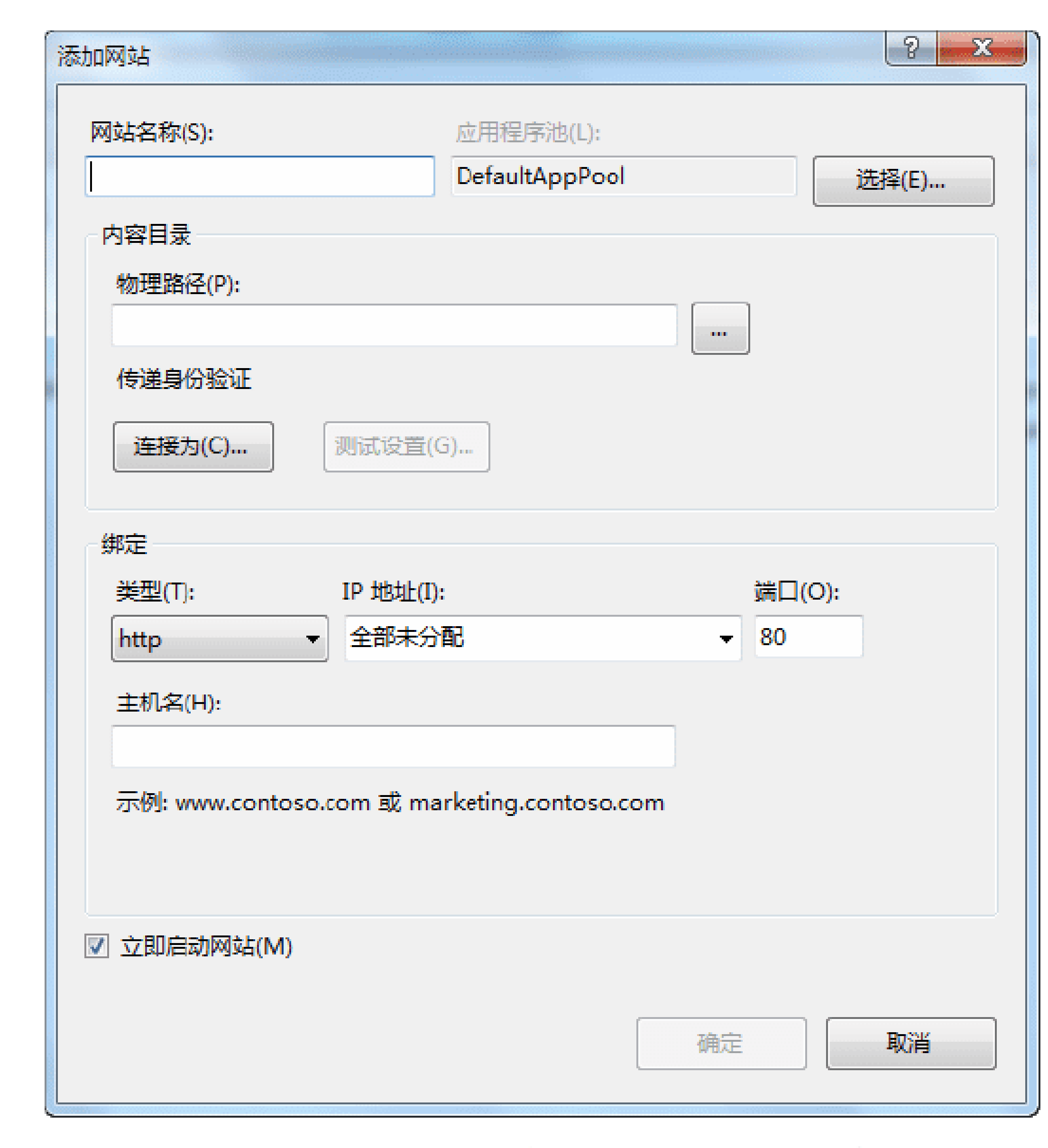


图 2-24　“添加网站”对话框

(3) 必须填写的是“网站名称”和“物理路径”，另外必须保证 IP 地址和端口号不和已有网站重复。在“绑定”部分，既可以填写 IP 地址，也可以从下拉列表中选择某个 IP 地址。其中类型通常选择“http”选项，另外每个 Web 站点都具有唯一的、由三个部分组成的标识，用来接收和响应请求的分别是端口号、IP 地址和主机头名。因此，一台服务器上的各个站点间不允许出现三者完全相同的情况。默认的 IP 地址为“全部未分配”，默认的端口号是 80，默认的主机头为空，三者都可以修改。

注意：

IP地址一栏可以指定Web站点的IP，但如没有特别需要，则选择“全部未分配”选项；如指定了多个主机头，则IP一定要选为“全部未分配”选项，否则访问者会访问不了网站。

(4) 在“内容目录”部分，输入新站点的主目录，或者利用边上的“浏览”按钮进行选择。

注意：

网站主目录必须是包含站点首页，如 index.html 或 default.html 等首页文件的目录。同时注意该目录的权限，必须具备一定的权限，否则该站点不可用。对于“连接为”按钮，是用于设置权限的，默认的是“应用程序用户”，当然也可以选择某个用户，但如果设置不合适也可能带来潜在的安全隐患。“测试设置”可以运行一个诊断过程，来发现可能存在的问题。如果该站点的配置是正确且勾选了“立即启动网站”选项时，单击“确定”按钮后则站点将自动启动。

使用上面提到的向导，用户已可以建立一个站点并做基本的配置，但如果希望对网站进行深入的配置，则需要进行更多的配置。

注意：

浏览器访问 IIS 时是按照如下顺序进行的：IP→端口→主机头→该站点主目录→该站点的默认首文档。正确安装和配置 IIS 之后，就已经有一个默认的站点了，该站点处于启动状态。因此，在新建网站时请注意如果全部使用默认值就可能与默认网站产生冲突，导致新站点不能正常启动。对此用户必须正确配置，且不能出现冲突。

2.4.3　IIS 的配置

站点建立以后，如果希望修改有关网站的配置，或者进行更多配置，可以在“Internet 信息服务(IIS)管理器”窗口左侧“网站”处右击所希望修改的站点进行配置。

这里可以进行的配置选项较多，总的来说有以下 3 个方面。

(1) 网站基本配置。可以为网站设置一个标识，配置 IP 地址和端口等；修改网站的主目录，设置 IIS 默认启动的文档。如果这些基本配置正确无误，网站启动后就可以在浏览器中进行验证。

(2) 网站性能配置。对网站访问的带宽和连接数进行限定，以更好地控制站点的吞吐量。如果是多站点服务器，通过对某个站点的带宽和连接数进行限制，这样可以放宽其他站点的访问量并提供更多的系统资源。在实际环境中，应该根据网络通信量和使用过程中的变化对性能参数做动态调整。

(3) 网站的安全性配置。为了保证 Web 网站和服务器的运行安全，可以进行“身份验证”、“IP 地址”、“域名”和“SSL”等方面的设置。

提示：

如果读者只是希望先建立一个简单的测试环境，进行简单的(静态)网页发布，则此时只需要修改主目录中的文件夹选项，将其指向所制作的网站根目录即可。

以下就 Windows 7 系统进行说明。

(1) 首先，在“控制面板”中打开“管理工具”，会看到如图 2-25 所示的显示界面，其中包含了“Internet 信息服务(IIS)管理器”的图标，双击即可打开。

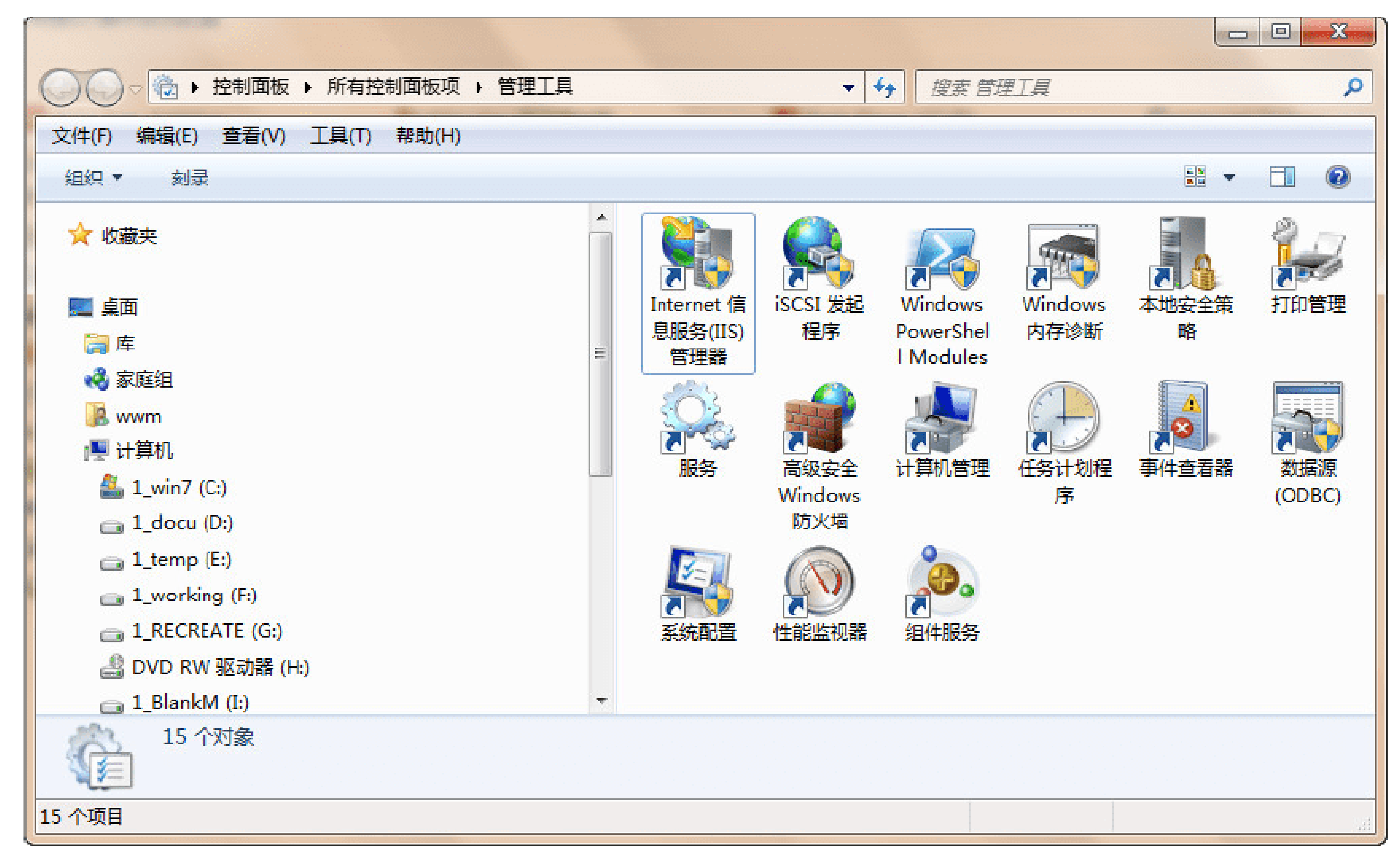


图 2-25　IIS 的配置方法

提示：

如果经常需要使用 IIS，建议将鼠标指到“Internet 信息服务(IIS)管理器”上，右击，在弹出的快捷菜单中选择“发送到”中的“桌面快捷方式”选项，这样就能从桌面直接进入 IIS，而不用每次都先进入控制面板。

(2) 双击上述的“Internet 信息服务(IIS)管理器”图标，即可打开 IIS 的管理器，如图 2-26 所示。

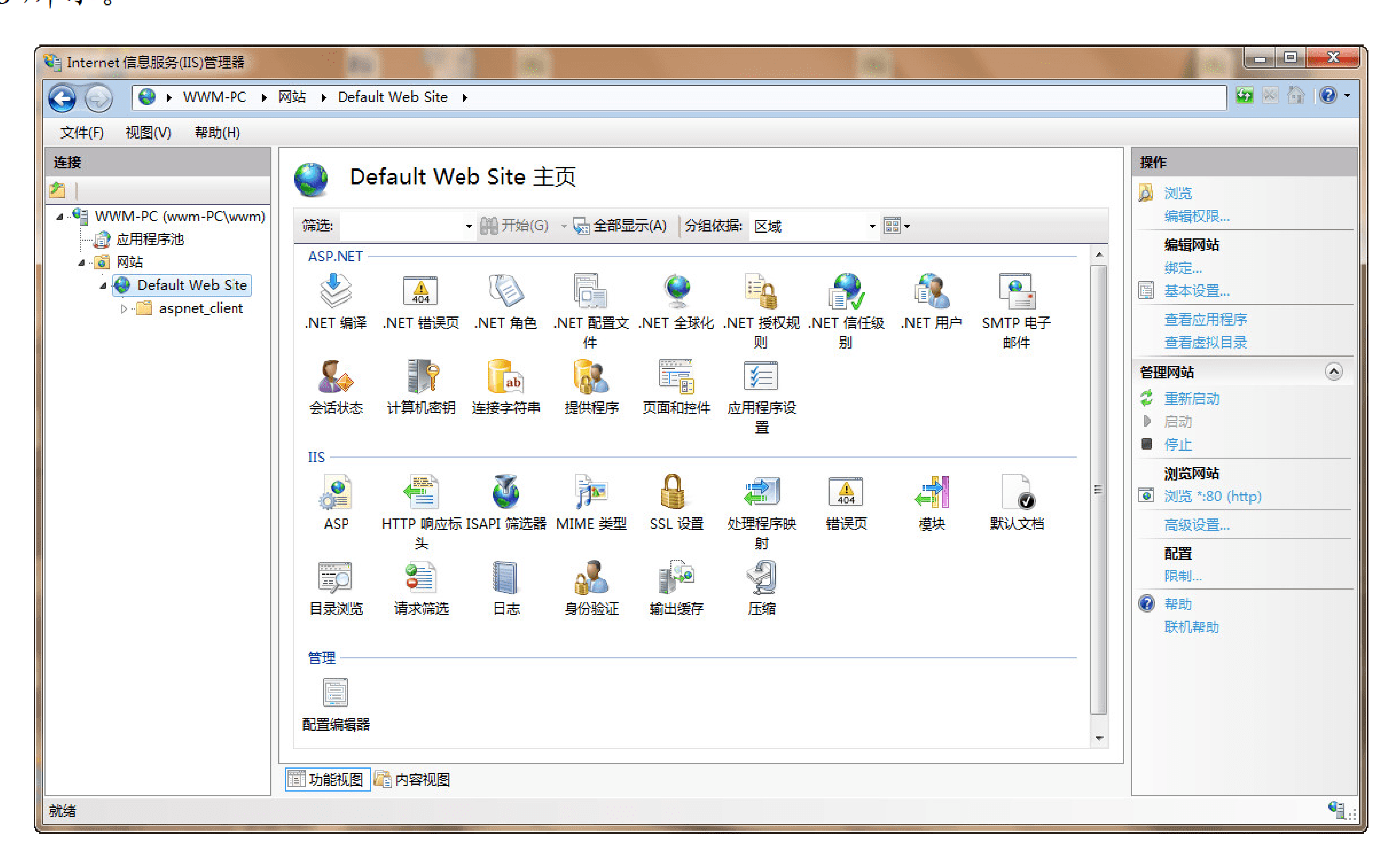


图 2-26　IIS 的配置界面

(3) 单击图 2-26 界面右侧的“高级设置”功能，即可打开如图 2-27 所示的配置界面，其中的“物理路径”需要设置为网站的根目录所在位置。

(4) 单击图 2-26 界面右侧的“绑定...”功能，即可打开如图 2-28 所示的配置界面，可以实现对主机名、端口、IP 地址等进行添加等的编辑操作。

(5) 如果已添加网站，则此时可以进入网站编辑状态，编辑界面如图 2-29 所示。

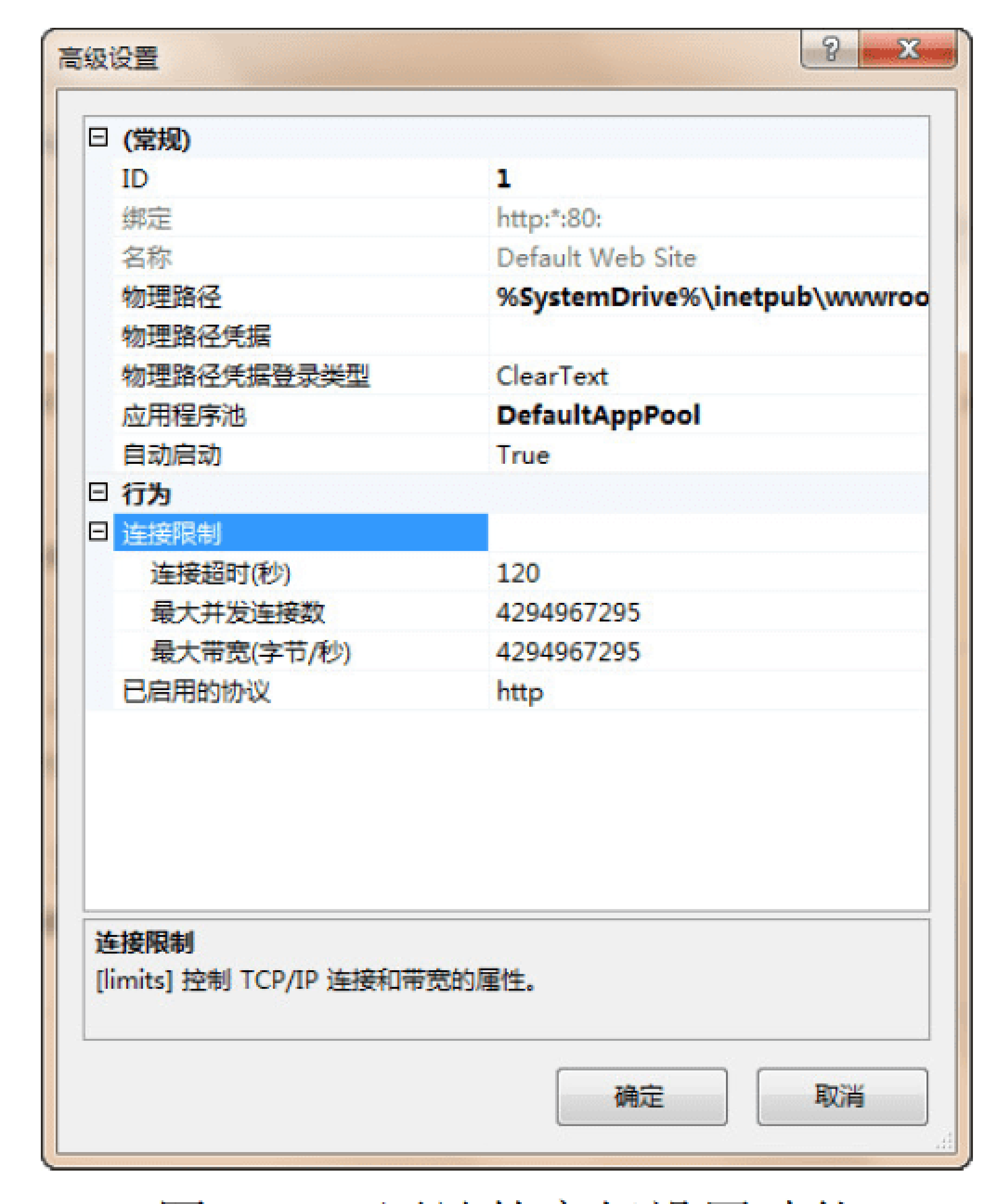


图 2-27　网站的高级设置功能

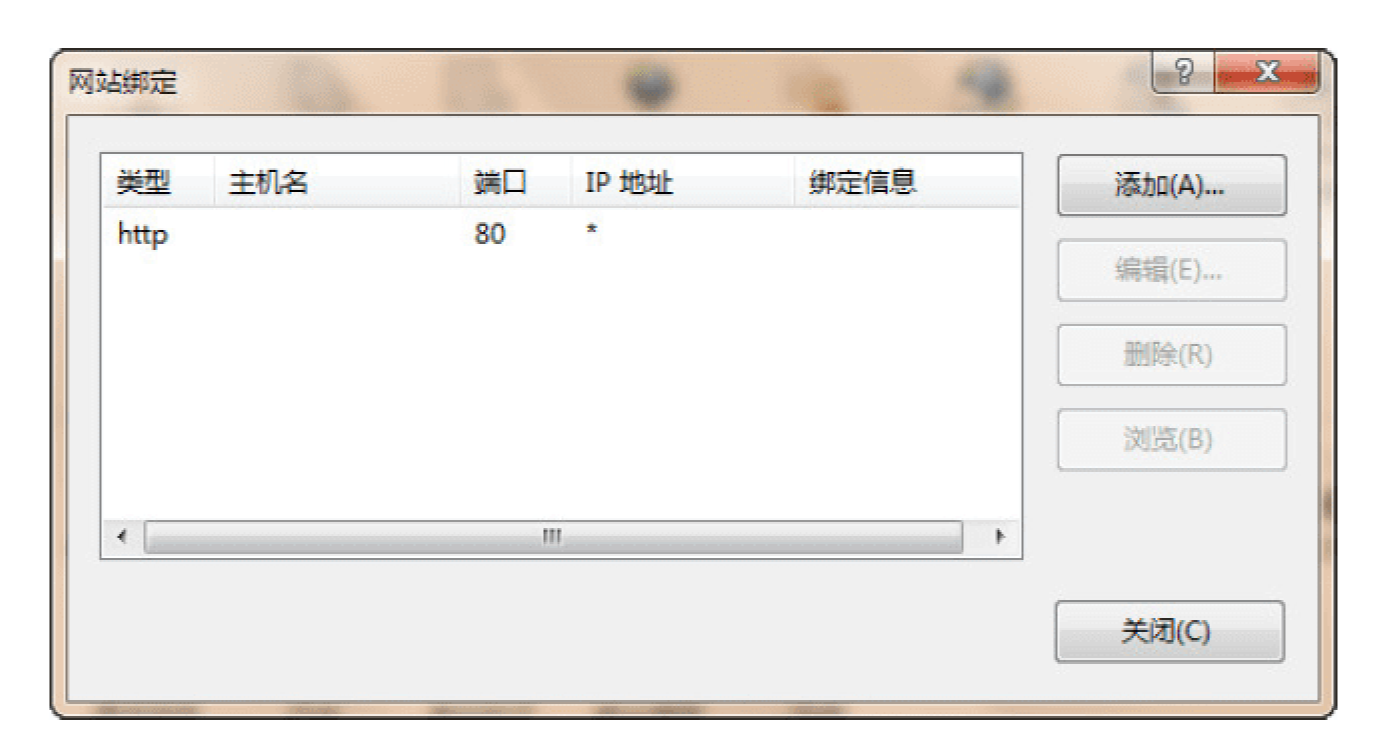


图 2-28　网站绑定设置功能

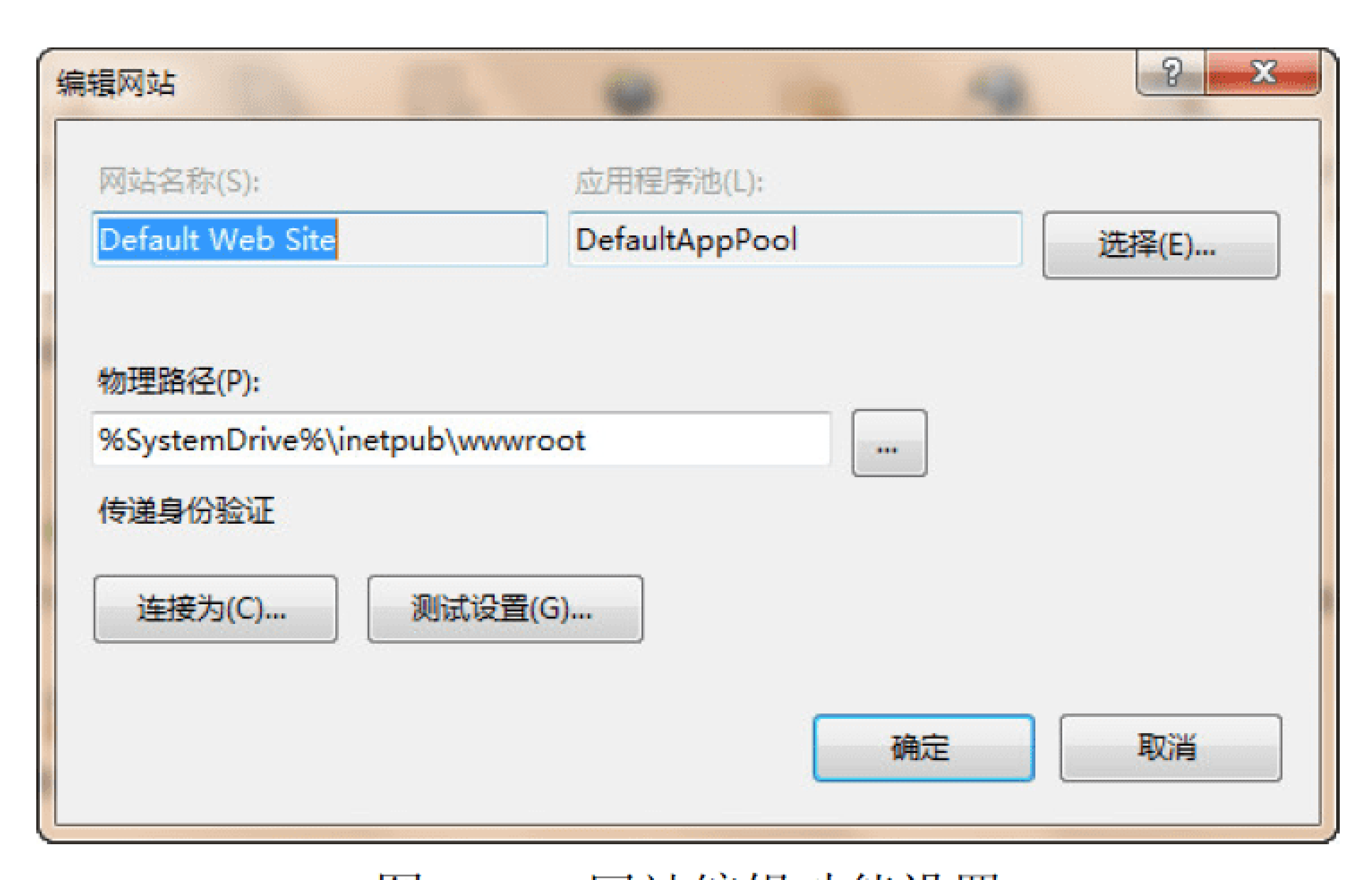


图 2-29　网站编辑功能设置

(6) 如果网站设置正确，则此时在浏览器地址栏中输入 “http://localhost/” 或者输入本机的 IP 地址就可以浏览该网站了，编辑界面如图 2-28 所示。若用户没有修改网站的物理路径，则此时会出现 IIS 默认的测试网站，如图 2-30 所示。



图 2-30　浏览测试网站

注意：

IIS 有更多的配置选项，因为篇幅限制，在此就不一一列举了，读者可以参考微软的有关文档进行进一步的配置。

2.4.4　其他 Web 服务器

不同的 Web 服务器具有不同的服务特性，包括支持的技术、性能、效率和支持的操作系统等，选择合适的 Web 服务器能提高网站的总体性能和服务品质。以下是一些其他常用的 Web 服务器。

1. Apache

除了 IIS 以外，Apache 是世界使用排名第一的 Web 服务器软件，超过 50%的网站在使用 Apache，它是以高效、稳定、安全、免费而著称的最受欢迎的服务器软件，并且可以运行在几乎所有广泛使用的计算机平台上。Apache 源于 NCSAhttpd 服务器，经过多次修改，成为世界上最流行的 Web 服务器软件之一。Apache 取自“a patchy server”的读音，意思是充满补丁的服务器，因为它是自由软件，所以不断有人来为它开发新的功能、新的特性和修改原来的缺陷。

它属于开放源代码的 Web 服务器软件，但是 Apache 对 ASP 或.NET 支持的不好，如果网站采用了这些技术方案，建议使用 Windows Server + IIS 来建构 Web 服务器。

2. GFE/GWS

GFE/GWS 是 Google 的 Web 服务器，目前用户数量正在不断增加。

3. Nginx

它不仅是一个小巧且高效的 HTTP 服务器，也可以做一个高效的负载均衡反向代理，通过它接受用户的请求并分发到多个 Mongrel 进程可以极大地提高 Rails 应用的并发能力。

4. Lighttpd

其是由德国人 Jan Kneschke 领导开发的，基于 BSD 许可的开源 Web 服务器软件，其根本的目的是提供一个专门针对高性能网站，安全、快速、兼容性好并且灵活的 Web Server 环境。具有非常低的内存开销，CPU 占用率低，效能好，以及丰富的模块等特点。Lighttpd 是众多 OpenSource 轻量级的 Web Server 中较为优秀的一个。支持 FastCGI、CGI、Auth、输出压缩(output compress)、URL 重写、Alias 等重要功能。

5. Zeus

其是一个运行于 UNIX 下的非常优秀的 Web Server，据说性能超过 Apache，是效率最高的 Web Server 之一。

6. BEA WebLogic

其是用于开发、集成、部署和管理大型分布式 Web 应用、网络应用和数据库应用的 Java 应用服务器。将 Java 的动态功能和 Java Enterprise 标准的安全性引入大型网络应用的开发、集成、部署和管理之中。 BEA WebLogic Server 拥有处理关键 Web 应用系统问题所需的性能、可扩展性和高可用性。

7. Tomcat

Apache 软件基金会(Apache Software Foundation)的 Jakarta 项目中的一个核心项目，由 Apache、Sun 和其他一些公司及个人共同开发而成。由于有了 Sun 的参与和支持，最新的 Servlet 和 JSP 规范总是能在 Tomcat 中得到体现。因为 Tomcat 技术先进、性能稳定，而且免费，因而深受 Java 爱好者的喜爱并得到了部分软件开发商的认可，成为目前比较流行的 Web 应用服务器。

2.5　网站运行的基础——安全

国际上计算机犯罪正以每年 100%的速度增长；网上的黑客攻击事件也大约以每年 10 倍的速度增长。自 1999 年计算机病毒首次被发现以来，其发展速度呈几何级数增长。据美国审计总署的资料，世界上 120 余个国家已经或正在研究进入计算机网络的手段；1995 年，入侵美国国防部计算机网络的事件多达 25 万次，其中 62.5%获得了成功，欧美等国金融机构的计算机网络被入侵的比例高达 77%。例如，2000 年 2 月 7 日至 9 日，美国著名的雅虎、亚马逊等 8 大网站接连遭受来历不明的攻击，导致服务系统中断，整个互联网使用率在此段时间内下降 20%，这次攻击给这些网站造成的直接损失达 12 亿美元，间接经济损失高达 10 亿美元。据美国 FBI 的调查，每年因网络安全造成的损失总额高达 170 亿美金；而 CERT 组织的数据表明，平均每五个站点就有一个遭受不同程度的攻击。我国近几年来计算机犯罪也以 30%的速度在增长。据有关部门统计，国内 90%以上的电子商务网站存在比较严重的安全漏洞，网络的安全问题正面临着日益严重的威胁，提高网站的安全性刻不容缓。

黑客攻击是对网站安全的最大挑战。虽然网站服务器管理人员都采取了多种防范措施，试图让自己的网站更安全，但黑客依然可以突破这些安全措施，攻入 Web 网站的内部窃取、篡改网站信息甚至造成服务中止。这往往是管理人员没有正确认识各种安全防范措施的作用，对网站的安全性做出了错误的评估。

2.5.1　网站安全威胁

首先网站是建立在操作系统上的，而操作系统是基于计算机硬件的，这样一个系统要对外提供服务还必须接入网络。网站也是使用程序员编写的各种软件构成的，客观的说，程序员编写的软件都是有可能存在漏洞的，有些漏洞也许要经过许多年后才被发现。所有

这些环节中的任何一个部分如果不能保证安全，就破坏了整体的安全性。

网站安全面临的主要威胁有信息截取、内部窃密和破坏、黑客、技术缺陷、计算机病毒和拒绝服务攻击等。

(1) 信息截取。信息截取指的是通过传输通道进行信息的截取，获取机密信息，或通过信息的流量分析、通信频度、长度分析等导出有用信息。这种方式不破坏信息的内容，不易被发现，在军事对抗、政治对抗和当今经济对抗中是最常用的，也是最有效的方式。问题在于计算机网络是建立在通信网络的基础上的，而多数专用通信网是以公网为基础的。虽然网络上有一些防止非授权用户进入网络的安全措施，但对熟悉网络软件和操作使用的攻击者来说这些措施是很脆弱的。

(2) 内部窃密和破坏。内部窃密和破坏是指内部或系统内部人员通过网络窃取机密、泄露、更改信息或破坏信息系统。美国 FBI 进行的一项调查表明，所有攻击中的 70%是从内部发动的，只有 30%来自于外部。

(3) 黑客。黑客(hacker)指技术上的行家或热衷于解决问题、克服限制的人。骇客(cracker)是那些喜欢进入他人系统的人。两者之间最主要的不同是：黑客们创造新东西，骇客们破坏东西，现在人们倾向于统一使用“黑客”指代这两种人。据统计，目前在互联网上至少有 4 万多个黑客网站，它们介绍的基本攻击手段超过 800 多种，而且手法也在不断翻新，其中银行、金融和证券机构等成为攻击的重点。

(4) 技术缺陷。首先，由于认识能力和技术发展的局限性，在硬件和软件设计过程中，难免留下技术缺陷，由此形成了网络的安全隐患。其次，网络硬件、软件产品多数依靠进口，为数不少的服务器都安装了微软的 Windows 操作系统，其中或多或少包含漏洞和后门。除了充分利用操作系统的漏洞之外，各种主流应用软件的漏洞也开始被利用，仅在 2006 年 3 至 5 月，Macromedia Flash、Microsoft Word、Apple QuickTime、iTune、Winamp 等知名软件纷纷被发现存在可被黑客利用的漏洞。

(5) 计算机病毒。自从 1988 年第一例病毒(蠕虫病毒)成功侵入美国军方网络，导致 8500 台计算机染毒和 6500 台计算机停机，造成直接经济损失近 1 亿美元之后，这类事件此起彼伏，据美国计算机安全协会(NCSA)最近的一项调查发现，几乎 100%的美国大公司都曾在他们的网络或台式机上经受过计算机病毒的危害。另据 2006 年 7 月 20 日瑞星公司发布的《中国大陆地区 2006 年上半年电脑病毒疫情和互联网安全报告》显示，2006 年上半年被截获的病毒共有 119402 个，比去年暴增了 5 倍，说明新病毒不断加速出现。从 2001 年的红色代码、2003 年的 SQL 蠕虫与冲击波、2006 年的“灰鸽子”及“熊猫烧香”等病毒传播和发作的情况看，计算机病毒感染方式已与黑客攻击手段紧密结合，利用网络、网站、操作系统的漏洞以及 U 盘进行传播，病毒“偷、骗、抢”等行为愈演愈烈、勒索木马开始流行。

(6) 拒绝服务攻击。拒绝服务攻击(Denial of Service，简称 Dos)是一种简单而有效的攻击方式，它利用大量非正常的请求并拒绝服务器所提供的服务来破坏网站的正常运行，最终使部分 Internet 连接和网络通信中断。这种攻击所衍生的方式有很多种，其中最基本的是利用合理的服务请求来占用过多的服务资源，从而使合法用户无法得到服务。可分为 SYN 洪水、SYN-ACK 洪水、UDP 洪水等。一种基本攻击过程为：首先攻击者向服务器发

送众多的带有虚假地址的请求，服务器发送回复信息后等待回传信息，由于地址是伪造的，所以服务器一直等不到回传的消息，分配给这次请求的资源就始终没有被释放。当服务器等待一定的时间后，连接会因超时而被切断，此时攻击者会再度传送新的一批请求，在这种反复发送伪地址请求的情况下，服务器资源最终会被耗尽。另外一种分布式的拒绝服务攻击(Distributed Denial of Service，简称 DDos)，是一种分布、协作的大规模攻击方式，主要瞄准比较大的站点，如商业公司、搜索引擎和政府部门的站点等。发动 DoS 攻击只要有一台联网的单机就可实现，与之不同的是 DDoS 攻击是利用一批受控制的计算机向一台计算机发起攻击，这种来势迅猛的攻击令人难以防备，因而往往具有更大的破坏性。

2.5.2　防范策略

网站的建设是需要大量投入的，但忽视安全因素，将可能使全部的努力付之东流，因此对网站的安全防护问题也应受到重视。网络安全技术包括：操作系统安全、加密、防火墙、安全认证、反病毒、入侵检测、安全扫描工具等。

(1) 操作系统安全。操作系统是介于计算机和网络之间的工作平台，从终端用户的程序到服务器应用服务以及网络安全的软件都是运行在操作系统上的。因此，保证操作系统的安全是整体安全的基础，除了不断安装安全补丁之外，还需要建立一套系统的监控机制，建立和实施有效的用户口令和访问控制等方面的制度。

(2) 加密。为了减少网站传输过程中的信息被截取后的危害，网站管理人员一般都采取 SSL 或其他加密的方法。当网站启用加密后，该网站发送和接收的信息都经过加密处理，在信息的传送过程中为信息提供加密保护。但是单靠加密还是无法保障网站的安全的，通过 SQL 注入、跨站脚本攻击等各种其他手段仍然可以对 Web 站点进行攻击。这就是为什么许多网站虽采用了 SSL 加密，但还是被黑客攻破的原因。

(3) Web 服务器安全。Web 服务器本身多是安全的。但是在编写网页代码时，不安全的脚本漏洞往往会成为黑客攻击的对象。通过使用网页即设计中的各种漏洞进行攻击，最终可以达到控制网站或修改网站内容等目的。因此，在设计网页中，对于涉及 Web 服务器及相关服务的命令要特别注意。

(4) 防火墙。在网站服务器上安装防火墙是保护自己的好办法，它能有效地防范多种攻击，对于网站安全来说是必需的。防火墙建立在通信技术和信息安全技术之上，用于在内外网之间建立一个安全屏障，根据指定的策略对网络访问并进行数据过滤、分析和审计。它主要用于 Internet 接入和专用网与公用网之间的安全连接。防火墙主要有病毒防火墙、包过滤防火墙、应用网关防火墙和代理服务器。

- 病毒防火墙能阻止部分外来病毒的侵袭，它主要通过检测外来数据来对付病毒的入侵。但由于病毒的种类繁多，加之可以进行拆包传输(即将病毒数据拆成多个小包)。因此，很难做到完全防范。
- 包过滤防火墙是在网络层中对数据包实施有选择的通过，根据系统内事先设定的过滤逻辑，检查数据流中每个数据包，通过数据包的源地址、目的地址、所用的端口及 TCP 链路状态等来确定是否允许该数据包通过。

- 应用网关防火墙是建立在网络应用层上的一种协议过滤器。针对特别的网络应用服务协议和数据进行过滤，它能够对数据包分析并形成报告。应用网关防火墙能对某些易于登录和控制所有输出/输入的通信环境给予严格的控制，以防有价值的程序和数据被窃取。
- 代理服务器是设置在 Internet 防火墙网关的专用应用级代理。这种代理服务允许网管准许或拒绝特定的应用程序或一个应用的特定功能。包过滤防火墙和应用网关防火墙都是通过特定的逻辑判断来决定是否允许特定的数据包通过，一旦判断条件满足，防火墙内部网络的结构便暴露在外来用户面前；而代理服务器可使防火墙内外计算机系统应用层的“链接”由两个终止于代理服务的“链接”来实现，从而成功实现了防火墙内外计算机系统的隔离。除此之外，代理服务还可用于实施较强的数据流监控、过滤、记录和报告等。代理服务器可由计算机硬件(如工作站)来承担。

防火墙有访问过滤机制，通过设置防火墙的“访客名单”，通过记录所有善意的访问者，把恶意访问排除在外，只允许善意的访问者进来，但还是无法应对许多恶意行为。因为，一个伪装成善意访问者的黑客的访问一旦被允许，后续的安全问题就不是防火墙能应对的了。如何鉴别善意访问和恶意访问就成了一个问题。而且，用来架构服务器的操作系统、服务器软件都可能有现在还未发现的漏洞，对此，一般防火墙是没有办法的。

(5) 安全认证。安全系统的建立都依赖于系统用户之间存在的各种信任关系。可靠的信息确认技术应具有：合法身份的用户可以校验所接收的信息是否真实可靠，并且能确认发送方是谁；发送信息者必须是合法身份用户，任何人不能冒名顶替伪造信息；出现异常时，可由认证系统进行处理。目前，信息确认技术已较为成熟，如信息认证、用户认证和密钥认证、数字签名等，这为信息安全提供了可靠保障。

(6) 反病毒、防木马。计算机病毒实际上就是一种在计算机系统运行过程中能够实现传染和侵害计算机系统的功能程序。在非法进入系统或违反授权的攻击成功后，攻击者通常要在系统中植入后门，为以后的攻击提供方便，如向系统中侵入病毒、蠕虫、木马或通过窃听、冒充等方式来破坏系统正常工作。对此，要提高防范意识，做到：对所有软件必须经过严格审查，经过查毒、杀毒后再使用；采用病毒实时防护软件和网络防火墙，定时地对系统中的所有工具软件及应用软件进行检测。

(7) 入侵检测。入侵检测系统是近年出现的网络安全技术，它不仅能提供实时的入侵检测，并且能使用跟踪、恢复或者断开网络连接等方式及时有效地对网络进行保护，它不仅用于阻止外部黑客的入侵，也能有效地防范来自内部的攻击。

(8) 安全扫描工具。安全扫描工具主要扫描网络服务器，它可以主动寻找系统的安全漏洞和薄弱环节，分析安全隐患并利用模拟攻击来测试系统的安全程度和防御能力，主要用于测试和评估系统的安全性，防患于未然。

(9) 勤于备份。如果以上的种种策略均告失败，且网站的重要数据和文件已经被破坏，在这种情况下，一个之前的备份就能发挥重要的作用，这是系统得以快速恢复的必要前提。在查明安全问题并解决后，恢复所备份的网站资料可以将损失降至最低。

(10) 用户审计。服务器应根据对用户和系统中的使用行为和事件进行详细的日志记录和审计，通过这些日志记录，做阶段性的审计(时间间隔应该设定为较小，如每天)，从而做到发现用户账号的盗用、恶意使用等问题，并便于尽早进行处理。

(11) 建立良好、可操作的安全制度。网站安全管理就是通过保证维护网站的机密性、完整性和可用性来管理和保护组织的所有信息安全的一项体制。通过安全管理，把具有信息安全保障功能的软硬件设施和管理以及使用信息的人整合在一起，以此确保整个网站达到预定程度的安全等级。建立网站安全管理体系可以强化员工的网站安全意识，规范组织安全行为；对组织的关键资源进行全面系统的保护，维持竞争优势；在网站受到侵袭时，确保业务持续开展并将损失降到最低。常见的网络信息安全管理体系一般有四个组成部分，首先是总体方针，其次是安全管理组织体系，第三是涵盖物理、网络、系统、应用、数据等方面的统一安全策略，第四是可操作的安全管理制度、操作规范和流程。

总的来说，为了更好地保护 Web 网站安全，需正确认识各种安全措施的功能特点，然后将它们整合为一个整体，形成立体、多重的防御体系。

2.6　本章小结

本章首先介绍了网站规划和设计，它是网站获得成功的前提，前期规划中较小的偏差往往会导致到实现阶段出现的错误得到放大的效果，因此必须仔细抉择，而这一环节却往往被没有经验的网站建设者所忽视。本章还介绍了后继章节中需要的 Web 服务器软件的安装与配置方法，最后介绍了能保证网站稳定运行的安全问题。

2.7　思考和练习

1. 网站策划的步骤和每个步骤的要求是怎样的？
2. 如何判断本机是否已经安装过 Web 服务器？如果没有，需要如何进行安装？
3. 使用 IIS 安装并设置了新网站后，再将网站的有关文件导入后是否就可以立即投入使用？
4. 网站安全的基本原则及设置方法有哪些？

第3章　HTTP协议及其开发与HTML语言基础

HTTP 协议提供了从 WWW 服务器到本地浏览器的超文本传输协议，它规定了 Web 交互的通信协议；而 HTML 以一种非线性的网状逻辑结构来组织文档，它具体规定了传输消息中资源实体的格式和类型等。本章旨在让读者了解 HTTP 的基本原理，掌握 HTTP 消息的类型和一般格式、HTML 的标签、文档结构和基本语法。

本章要点：

- 理解 HTTP 的基本原理及运行机制
- 了解 HTTP 应用开发的基本方法
- 掌握 HTML 的标签、文档结构和基本语法

3.1　HTTP 协议

3.1.1　HTTP 概述

当用户想浏览一个网站时，首先需要在浏览器的地址栏里输入网站地址，如 www.sohu.com，但单击 Enter 键后浏览器地址栏里出现的却是 http://www.sohu.com，其中的“http://”是浏览器自动添加的，它代表了什么呢？

我们将用户在浏览器的地址栏里输入的地址称为 URL(Uniform Resource Locator，即：统一资源定位符)。就如同每家每户都有一个门牌一样，每个网页也都有一个 Internet 地址。当你在浏览器的地址栏中输入一个 URL 或是单击一个超级链接时，URL 就确定了要浏览的地址。浏览器通过超文本传输协议(HTTP)，将 Web 服务器上站点的网页代码提取出来，并翻译成网页。

Internet 的基本协议是 TCP/IP 协议，在 TCP/IP 上层的是应用层(Application Layer)，它包含所有高层的协议。这些高层协议包括文件传输协议 FTP、电子邮件传输协议 SMTP、域名系统服务 DNS、网络新闻传输协议 NNTP 和 HTTP 协议等。

自 WWW 诞生以来，一个丰富多彩的虚拟世界便出现在我们眼前，用户怎样才能在这个浩瀚的海洋中快速获取其所需要的信息呢？自从采用超文本作为 WWW 文档的标准格式后，1990 年有关专家制定了能够快速定位并传输这些超文本文档的协议，即 HTTP 协议。经过若干年来的使用与发展，它得到不断的完善和扩展，目前正在使用的是 HTTP/1.0 的

第六版。

HTTP 协议(Hypertext Transfer Protocol，即：超文本传输协议)是用于从 WWW 服务器传输超文本到本地浏览器的传送协议，它是分布式 Web 应用的核心技术协议，在 TCP/IP 协议栈中属于应用层，更详细的内容可以参考 RFC2616。其中定义了 Web 浏览器向 Web 服务器发送索取 Web 页面请求的格式，以及 Web 页面在 Internet 上的传输方式。它不仅保证计算机正确快速地传输超文本，还确定先传输文档中的哪一部分，以及哪部分内容首先显示(如文本先于图形)等。这就是所有用户在浏览器中所看到的网页地址以“http://”开头的原因。

HTTP 首次出现在 1990 年，即 HTTP 0.9 版本。它适用于各种数据信息的简洁快速协议，但是远不能满足日益发展的各种应用的需要。HTTP 0.9 作为 HTTP 协议具有典型的无状态性：每个事务都是独立进行处理的，当一个事务开始后，就在客户与服务器之间建立一个连接，当事务结束时才释放这个连接。随后，出现了 HTTP 1.0 版本。基本协议是无状态的，说明客户机和服务器在会话期间不存储关于对方的信息。客户机和服务器连接，服务器传输请求的信息，然后连接关闭。服务器不必知道关于客户机的任何信息，它只提供请求的信息。

HTTP 1.0 成为最重要的面向事务的应用层协议。在该版本中，Web 页上的每个对象(如图像)均要求建立一个新的连接以传输该对象。它的特点是简单、易于管理，由于它符合大多数用户的需要，因此得到了广泛的应用。存在的缺点是对用户请求响应较慢、网络拥塞严重、安全性较低等。在 HTTP 1.1 版本中增加了连续性，连续性允许客户机和服务器保持连接(不为状态所迷惑)，直到将一个 Web 页上的所有对象传输完毕，最后才关闭连接，这使客户机/服务器之间的连接更为高效。

HTTP 1.1 版本支持浏览器的高速缓冲存储器管理。通常一个服务器页对应于一个浏览器高速缓冲存储器中的一页，请求时就只需发送需要更新的项。这样既可以减少时间延迟，又节省了带宽。测试表明 HTTP 1.1 使下载次数减少了大约 50%，且减少了超过 50%的数据分组的数量。HTTP 1.1 还包括一个功能，在 HTTP 标题中的一个字段允许分配给单个 IP 地址多个域名，这样就有效缓解了 IP 地址即将逐渐被耗尽的问题。此外，HTTP 1.1 服务器端处理请求时按接收到的顺序来进行，这保证了传输的正确性。当然，服务器端在发生连接中断时，会自动重传请求，确保数据的完整性，目前大多数 Web 服务器和 Web 浏览器已使用了 HTTP 1.1 版。总的来说，HTTP 1.1 有如下几个特点：

- 能够识别主机名，允许多个虚拟主机名共存于一个 IP 上。
- 具有内容协商的能力，允许服务器以多种格式存取资源，供 Web 服务器和 Web 浏览器选择最佳版本。
- 通过持续性连接，加速 Web 服务器的响应速度。
- 允许 Web 浏览器请求索取文件的某部分，支持断点续传功能。

HTTP 1.2 版本，在资源分级上得到了更强有力的支持，同时对文本菜单界面的支持也更好，这适合于移动客户端等计算环境。作为设计目标的一部分，其在功能上更像是一个只读的全球网络文件系统。系统包含一系列层次性、可链接的菜单，菜单项与标题的选择

是由服务器管理员控制的。其中一些新特性已经成为 Gopher 协议的一部分，该协议主要面向菜单——文档设计，并且是 WWW 的先驱，相比 HTTP 1.1 版本其主要的改进是：使用了 SRV records 以更好地支持负载平衡，并且对于 Web 和 E-Mail 来说只会使用域名；改进了 Basic 和 Digest 访问认证，相比于之前的基于表单的认证，提供了更好的具有本地观感的浏览器体验；增加了一套新的 accepted headers——与过去的方式完全不同，只要不处于 accepted headers 中的任何头都会被兼容的服务器拒绝掉；可以通过 IETF 站点增加新的 accepted headers，它会象征性地收取一定的费用来补偿管理上的花费。

HTTP 2.0 即超文本传输协议 2.0，是下一代 HTTP 协议，是由互联网工程任务组(IETF)的 Hypertext Transfer Protocol Bis (httpbis)工作小组进行开发。HTTP 2.0 在 2013 年 8 月进行首次合作共事性测试。在开放互联网上 HTTP 2.0 将只用于 https://网址，而 http://网址将继续使用 HTTP 1.0，目的是在开放互联网上增加使用加密技术，以提供强有力的保护去遏制主动攻击。DANE RFC 6698 允许域名管理员不通过第三方 CA 自行发行证书。HTTP 2.0 的设计目标包括异步连接复用、头压缩和请求反馈管线化并保留与 HTTP 1.1 的完全语义兼容。httpbis 工作小组最初考虑了 Google 的 SPDY 协议、Microsoft 的 SM 协议和 Network-Friendly HTTP 更新。Facebook 对各方案进行了评价并最终推荐了 SPDY 协议。HTTP 2.0 的首个草稿于 2012 年 11 月发布，其内容基本和 SPDY 协议相同。

3.1.2　HTTP 的宏观工作原理

HTTP 协议是基于请求/响应范式的。客户机与服务器建立连接后，就可以向服务器发送请求，请求的格式依次为：统一资源标识符、协议版本号、 MIME 信息(包括请求修饰符、客户机信息和可能的内容)。服务器收到请求后，响应信息的格式为：一个状态行(包括：信息的协议版本号、一个成功或错误的代码)、MIME 信息(包括服务器信息、实体信息和其他可能的内容)。

大多数的 HTTP 通信是由用户初始化的，且通常包含至少一个请求服务器资源的申请。HTTP 通信建立在 TCP/IP 连接之上，其默认端口为 TCP 80，这个端口用户是可以变更的。这意味着 HTTP 使用了一个可靠的传输。最简单的情况发生于用户代理(UA)和目标服务器(O)之间通过一个独占的连接来进行，如图 3-1 所示。整个过程类似于电话订货，客户首先打电话给商家，提出购货请求，然后商家回应是否有货，最后才进行交易。

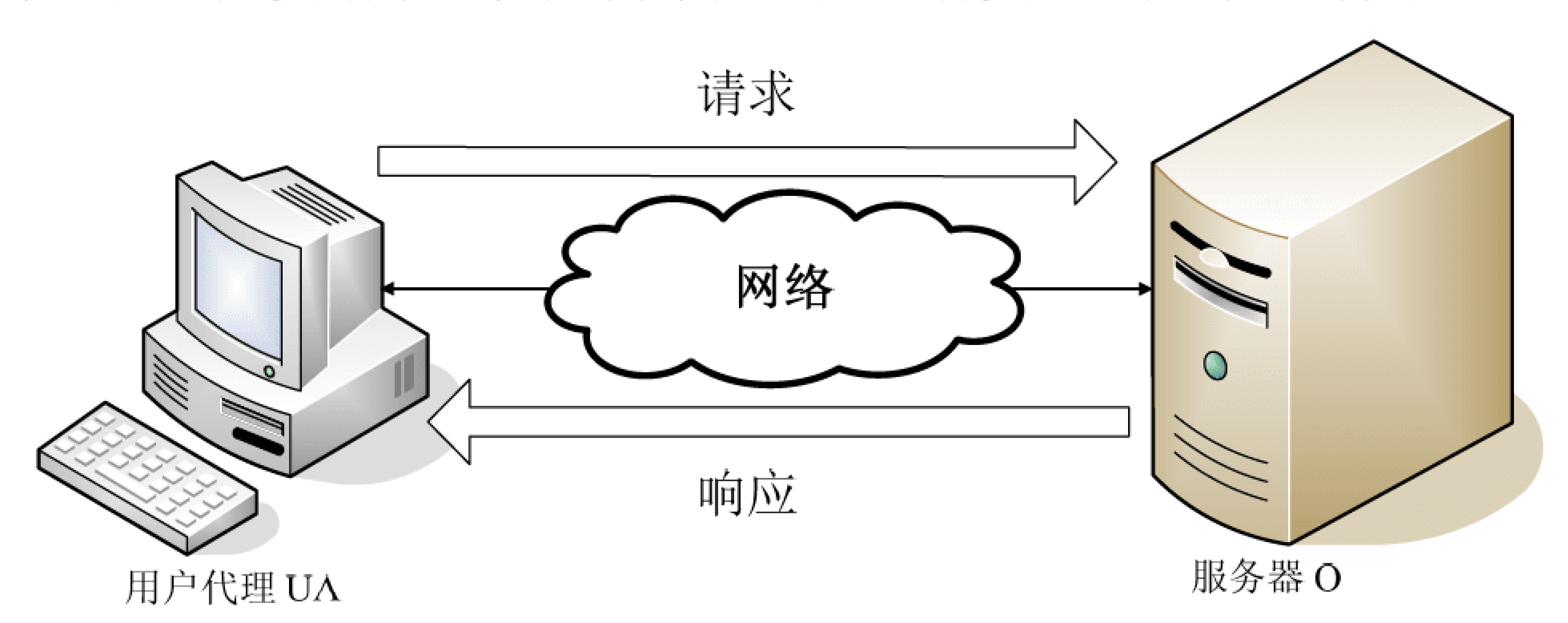


图 3-1　简单的 HTTP 通信

当一个或多个中介出现在请求/响应链中时，情况就变得复杂一些。中介有三种：代理(Proxy)、网关(Gateway)和隧道(Tunnel)。一个代理根据绝对格式来接受请求，重写全部或部分消息，通过请求的标识把已格式化过的请求发送到服务器。网关是一个接收代理，作为一些其他服务器的上层可以把请求翻译给下层的服务器协议。一个隧道是不改变消息的两个连接之间的中继点。当通信需要通过一个中介(如防火墙等)或者是中介不能识别消息的内容时，隧道经常被使用，这种通过中介的 HTTP 通信如图 3-2 所示。

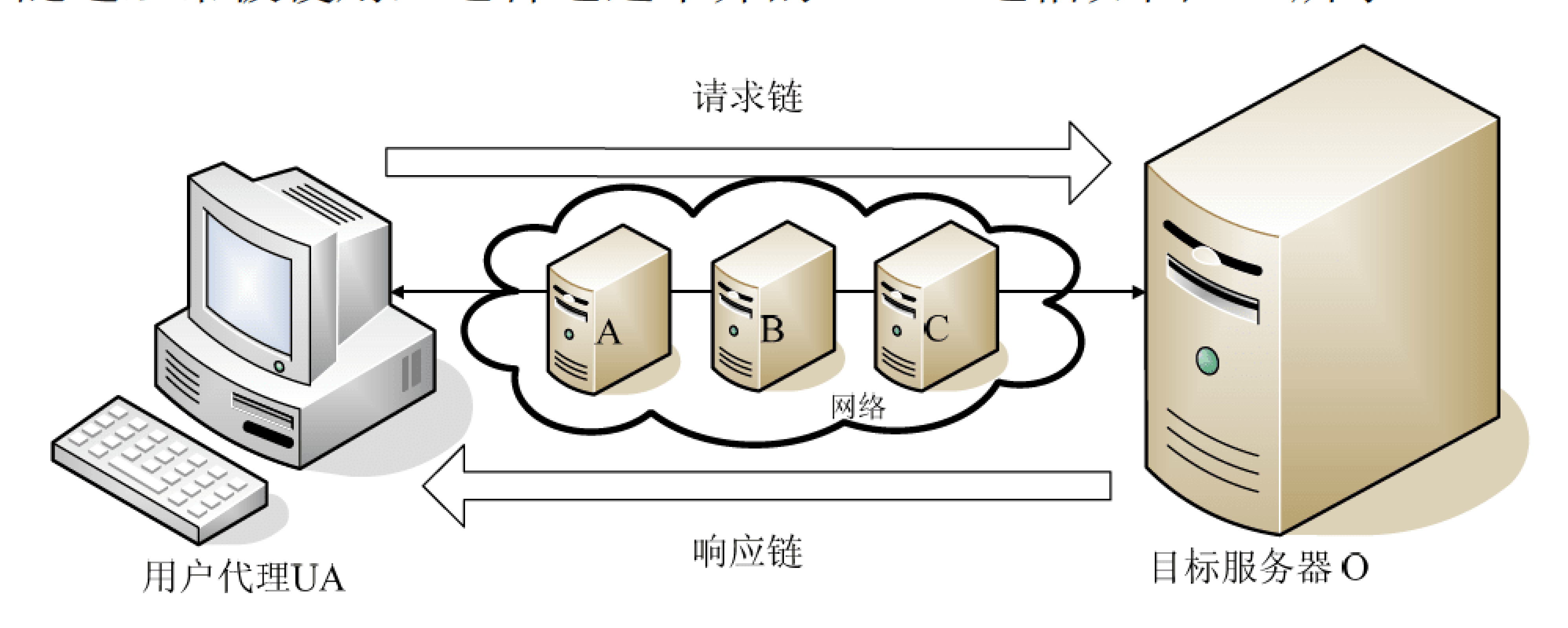


图 3-2　通过中介的 HTTP 通信

图 3-2 表明了在用户代理(UA)和目标服务器(O)之间有三个中介(A、B 和 C)。一个请求或响应消息必须经过四个连接段，相比之下有一些 HTTP 通信可能选择最近的连接、没有通道的邻居、应用在链的终点或沿此链的所有连接。尽管图 3-2 中所显示的是线性的，但实际上每个参与者都可能是多重的、并发的通信，例如：B 可能从许多客户机接收请求而不通过 A，并且/或者不通过 C 把请求发送到 A，同时它还可能处理 A 的请求。

上述环节中的任何一个节点都可以为提高处理效率而启用内部缓存。使用缓存的必要条件是沿链的参与者之一具有针对特定请求的缓存，其最终效果是请求/响应链被缩短为从用户代理该节点为止。图 3-3 说明这种通过带缓存功能中介的 HTTP 通信过程，在此次通信过程中，A 没有请求的缓存或没有启用缓存，而 B 有一个经过 C 来自 O 的前期响应的缓存拷贝，因此实际的通信过程到 B 为止。

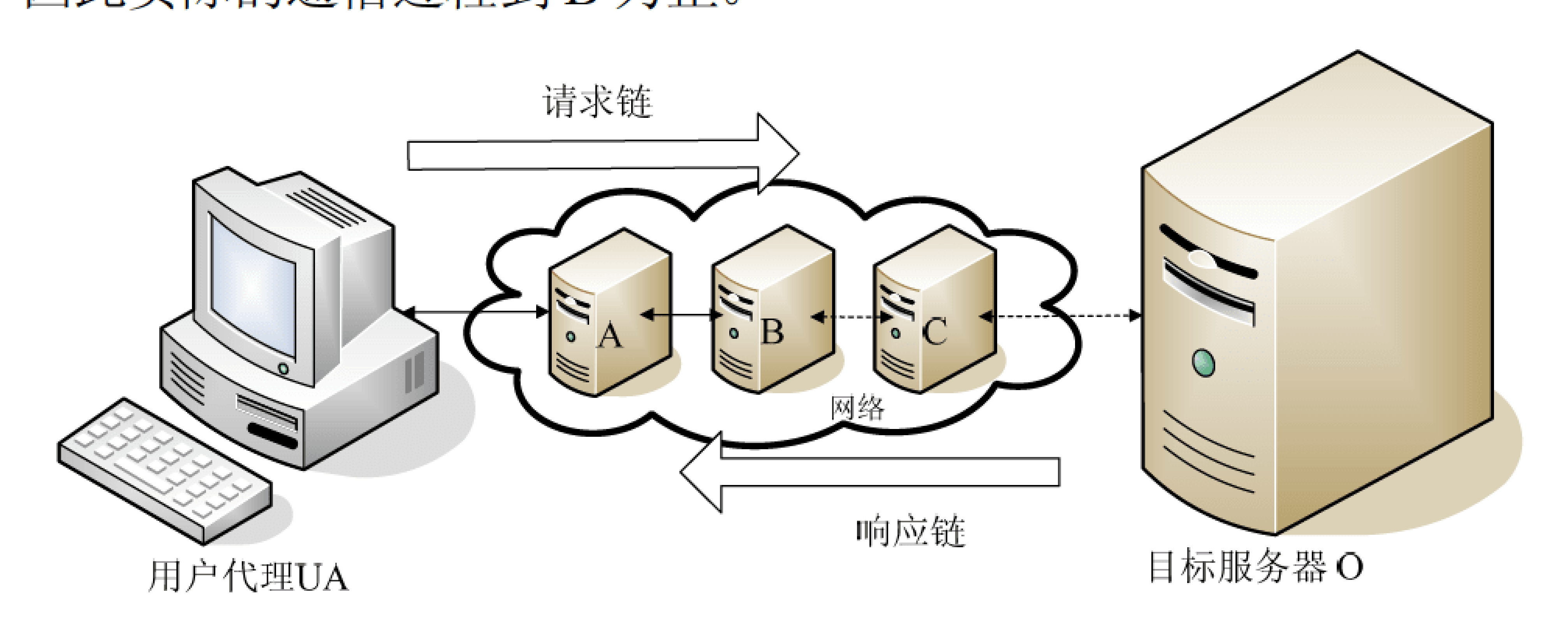


图 3-3　通过带有缓存功能中介的 HTTP 通信

HTTP 协议具有无连接/有连接、无状态的特点。

- 无连接/有连接：早期的 HTTP 是一个无连接的协议，其中无连接指的是限制每次连接只处理一个请求。客户和服务器连接后提交一个请求，在客户收到应答后立即断开连接。采用这种“无连接”协议，在没有请求提出时，服务器就不会在那

里空闲等待。完成一个请求之后，服务器不会继续再为这个请求负责，从而不用为了保留请求历史而耗费宝贵的资源。因此采用这种方式可以节省传输时间。但作为改进，后期版本的 HTTP 采用了一种可持续连接的方式。

注意：

HTTP 0.9 和 HTTP 1.0 使用非持续连接：限制每次连接只处理一个请求，服务器处理完客户的请求，并收到客户的应答后，即断开连接。采用这种方式可以节省传输时间。为了提高通信的效率，HTTP 1.1 之后使用了持续连接：不必为每个 Web 对象创建一个新的连接，一个连接可以传送多个对象。

- 无状态：HTTP 协议是无状态协议。无状态是指协议对于事务处理没有记忆能力，客户端只需要简单地向服务器请求下载某些文件，无论是客户端还是服务器都没有必要记录彼此过去的行为，每一次请求之间都是独立的。当然，这既是优点也是缺点。一方面，由于缺少状态使得 HTTP 的累赘少，系统运行效率高，服务器应答快；另一方面，由于没有状态，协议对事务处理没有记忆能力，若后续事务处理需要有关前面处理的信息，那么这些信息必须在协议外面保存。另外，缺少状态意味着所需的前面信息必须重现，导致每次连接需要传送较多的信息。为弥补该缺陷，可采用 cookie 技术和 session 技术作为在客户端与服务器之间保持状态的技术解决方案。

3.1.3　HTTP 协议基础

在 WWW 中，“客户”与“服务器”是相对的概念，它们仅存在于一个特定的连接期间，有时在某个连接中的客户在另一个连接中可能作为服务器。基于 HTTP 协议的客户/服务器模式的信息交换过程，可分为四个步骤：建立连接、发送请求信息、发送响应信息和关闭连接。

简单地说，任何服务器上除了存储有 HTML 文件以外，还有一个 HTTP 驻留程序，用于响应用户请求。用户的浏览器充当 HTTP 客户，向服务器发送请求，当用户在浏览器中输入了一个开始文件或单击了一个超级链接时，浏览器就向服务器发送了 HTTP 请求，此请求被送往该 URL 所指定 IP 地址的服务器。服务器端的驻留程序接收到该请求，在进行必要的操作后将所要求的文件送回。在这一过程中，在网络上发送和接收的数据已经被分成一个或多个数据包(packet)，每个数据包包括：要传送的数据以及控制信息(告诉网络怎样处理数据包)。而 TCP/IP 则决定了每个数据包的格式，实际上为了传输和处理的需要，信息是被分成许多小块然后再重新组合起来的。可以将服务器形象地类比为商家，商家除了拥有商品之外，还有一个职员在接听客户的电话，当客户打电话的时候，客户的声音转换成各种复杂的数据，通过电话线传输到商家的电话机，客户的电话机也负责将各种复杂的数据转换成声音，使得商家的职员能够明白客户的请求。这个过程客户是不需要明白声音是怎么转换成复杂的数据的。

基于 HTTP 协议的信息交换过程，可以用图 3-4 来表示，其一共分为四个步骤：建立

连接、发送请求信息、发送响应信息和关闭连接。

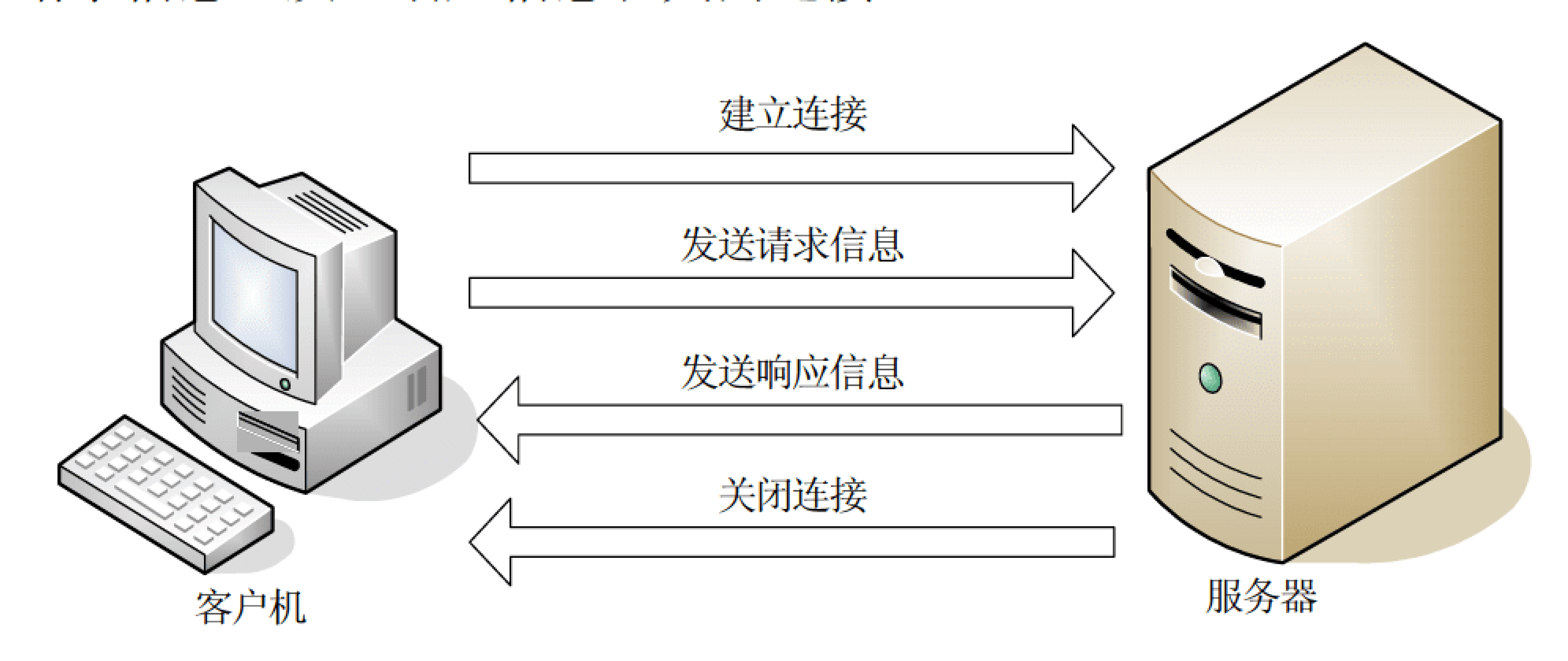


图 3-4　基于 HTTP 协议的信息交换过程

- 建立连接：连接的建立是通过申请套接字(Socket)来实现的。客户打开一个套接字并将它绑定在一个端口上，如果成功，就相当于建立了一个虚拟文件，以后就可以在该虚拟文件上写数据以通过网络向外传送。
- 发送请求信息：打开一个连接后，客户机把请求消息发送到服务器的特定端口上，完成提出请求动作。
- 发送响应信息：服务器在对客户的请求处理完毕后，要向客户机发送响应消息。
- 关闭连接：客户和服务器双方中的任何一方都可以通过关闭套接字来结束 TCP/IP 对话。

1. HTTP 请求消息

通常的 HTTP 消息包括两类：客户机向服务器发送的请求消息和服务器返回的响应消息，这两种类型的消息由一个起始行、一个或者多个头域、一个指示头域结束的空行和可选的消息体组成。其中请求消息的格式为：

请求消息=请求行(通用头|请求头|实体头)CRLF[实体内容]
请求行=方法　请求 URI　HTTP 版本号　CRLF
方法=GET|HEAD|POST|扩展方法
URI=协议名称+宿主名+目录与文件名

某个请求消息实例如下所示：

```
GET http://www.njupt.edu.cn/index.html HTTP/1.1
Host:www.njupt.edu.cn:80
Accept:*/*
Pragma:no-cache
Cache-Control:no-cache
```

上例第一行为请求行，它表示 HTTP 客户端(可能是浏览器、下载程序)通过 GET 方法获得所指定的 URI(此处为 http://www.njupt.edu.cn/index.html)所指向的文件，之后的几行使用了不同的通用头对本次请求进行了更详细的说明。

其中的方法表示对于“请求 URL”执行的方法，这个字段是大小写敏感的，可包括 OPTIONS、GET、HEAD、POST、PUT、DELETE、TRACE。方法 GET 和 HEAD 应该被

所有的通用 Web 服务器支持，其他所有方法的实现是可选的。GET 方法取回由“请求 URI”标识的信息。HEAD 方法也是取回由“请求 URI”标识的信息，只是可以在响应时，不返回消息体。POST 方法可以请求服务器接收包含在请求中的实体信息，可以用于提交表单，向新闻组、BBS、邮件群组和数据库发送消息。

“请求 URI”遵循 URI 格式，在此字段为星号(*)时，说明请求并不用于某个特定的资源地址，而是用于服务器本身。HTTP-Version 表示支持的 HTTP 版本，如为 HTTP/1.1。CRLF 表示换行回车符。以下对请求中的附加头信息进行简要说明。

(1) 通用头信息

通用头信息包含请求和响应消息都支持的头信息，通用头信息包含 Cache-Control、Connection、Date、Pragma、Transfer-Encoding、Upgrade 和 Via。对通用头信息的扩展要求通信双方都支持此扩展，如果存在不支持的通用头信息，一般将会作为实体头处理，常用的通用头信息如下。

① Cache-Control 头域。Cache-Control 指定请求和响应遵循的缓存机制。在请求消息或响应消息中设置 Cache-Control 并不会修改另一个消息处理过程中的缓存处理过程。请求时的缓存指令包括 no-cache、no-store、max-age、max-stale、min-fresh、only-if-cached，响应消息中的指令包括 public、private、no-cache、no-store、no-transform、must-revalidate、proxy-revalidate、max-age。各个消息中的指令含义如下。

- public：指示响应可被任何缓存区缓存。
- private：指示对于单个用户的整个或部分响应消息，不能被共享缓存处理。这允许服务器仅描述当前用户的部分响应消息，此响应消息对于其他用户的请求无效。
- no-cache：指示请求或响应消息不能缓存。
- no-store：用于防止重要的信息被无意地发布。在请求消息中发送将使得请求和响应消息都不使用缓存。
- max-age：指示客户机可以接收生存期不大于指定时间(以秒为单位)的响应。
- min-fresh：指示客户机可以接收响应时间小于当前时间加上指定时间的响应。
- max-stale：指示客户机可以接收超出超时期间的响应消息。如果指定 max-stale 消息的值，那么客户机可以接收超出超时期指定值之内的响应消息。

② Date 头域。Date 头域表示消息发送的时间，时间的描述格式由 rfc822 定义。例如，Date:Mon,31Dec200104:25:57GMT。Date 描述的时间表示世界标准时，如果希望换算为本地时间，需要知道当前用户所在的时区。

③ Pragma 头域。Pragma 头域用来包含实现特定的指令，最常用的是 Pragma:no-cache。在 HTTP 1.1 协议中，它的含义和 Cache-Control:no-cache 相同。

(2) 请求消息

请求头域允许客户端向服务器传递关于请求或者关于客户机的附加信息。请求头域可能包含下列字段 Accept、Accept-Charset、Accept-Encoding、Accept-Language、Authorization、From、Host、If-Modified-Since、If-Match、If-None-Match、If-Range、If-Range、If-Unmodified-Since、Max-Forwards、Proxy-Authorization、Range、Referer、User-Agent。

对请求头域的扩展要求通信双方都支持，如果存在不支持的请求头域，一般将会作为实体头域处理。

① Host 头域。Host 头域指定请求资源的 Internet 主机和端口号，必须表示请求 URL 的原始服务器或网关的位置。HTTP 1.1 请求必须包含主机头域，否则系统会以 400 状态码返回。

② Referer 头域。Referer 头域允许客户端指定请求 URI 的源资源地址，这可以允许服务器生成回退链表，可用来登录、优化 cache 等。它也允许废除的或错误的连接由于维护的目的被追踪。如果请求的 URI 没有自己的 URI 地址，Referer 不能被发送。如果指定的是部分 URI 地址，则此地址应该是一个相对地址。

③ Range 头域。Range 头域可以请求实体的一个或者多个子范围，正是有了这个选项才能实现断点续传功能。其使用方式如下。

表示头 500 字节：bytes=0−499

表示第二个 500 字节：bytes=500−999

表示最后 500 字节：bytes=−500

表示 500 字节以后的范围：bytes=500−

第一个和最后一个字节：bytes=0−0, −1

同时指定几个范围：bytes=500−600, 601−999

但是服务器可以忽略此请求头，如果无条件 GET 包含 Range 请求头，响应会以状态码 206(PartialContent)返回而不是 200(OK)。

④ User-Agent 头域。User-Agent 头域的内容包含发出请求的用户信息。

(4) 实体信息

实体信息一般由实体头域和实体组成。实体头域包含关于实体的原信息，实体头包括 Allow、Content-Base、Content-Encoding、Content-Language、Content-Length、Content-Location、Content-MD5、Content-Range、Content-Type、Etag、Expires、Last-Modified、extension-header。extension-header 允许客户端定义新的实体头，但是这些域可能无法被接收方识别。实体可以是一个经过编码的字节流，它的编码方式由 Content-Encoding 或 Content-Type 定义，其长度由 Content-Length 或 Content-Range 定义。限于篇幅，此处仅介绍 Content-Type 实体头和 Last-Modified 实体头。

① Content-Type 实体头。Content-Type 实体头用于向接收方指示实体的介质类型，指定 HEAD 方法发送到接收方的实体介质类型，或 GET 方法发送的请求介质类型。

Content-Range 实体头用于指定整个实体中的一部分的插入位置，也指示了整个实体的长度。在服务器向客户返回一个部分响应时，它必须描述响应覆盖的范围和整个实体长度。一般格式如下。

```
Content-Range:bytes-unitSPfirst-byte-pos-last-byte-pos/entity-legth
```

如欲传送前 500 字节/次字段的形式为：Content-Range:bytes0-499/1234。如果一个 HTTP 消息包含此节(例如，对范围请求的响应或对一系列范围的重复请求)，Content-Range 表示

传送的范围，Content-Length 表示实际传送的字节数。

② Last-Modified 实体头。Last-Modified 实体头指定服务器上保存内容的最后修订时间。

2. HTTP 响应消息

HTTP 1.0 的响应消息格式如下：

响应消息=状态行(通用信息头|响应头|实体头)　CRLF　〔实体内容〕
状态行=HTTP 版本号　状态码　原因叙述

响应头的信息包括：服务程序名，通知客户请求的 URL 需要认证，请求的资源何时能使用。

某个典型的响应消息如下：

```
HTTP 1.1 200 OK
Connectlon:close
Date:Mon,31Dec200104:25:57GMT
Server:Apache/1.3.14(Unix)
Content-type:text/html
Last-modified:Tue,17Apr200708:21:46GMT
Etag:"0d43c77040c01:4c8"
Content-length:453788
(数据…)
```

这个响应消息分为 3 部分：1 个起始的状态行(status line)、7 个头域、1 个包含所请求对象本身的附属体。上面的消息格式中，其中的“HTTP 版本号”表示支持的 HTTP 版本，此处为“HTTP 1.1”；状态码是一个三个数字的结果代码，主要用于机器自动识别，此处为“200”；原因叙述提供了一个简单的文本描述，用于帮助用户理解，此处为“OK”。此外的信息包括了通用信息头、响应头和实体头。本例的状态行表明，服务器使用 HTTP 1.1 版本，响应过程完全正常(也就是说服务器找到了所请求的对象，并正在发送)。

现在看一下本例中的各个头域。服务器使用 Connectlon:close 头域告知客户自己将在发送完本消息后关闭 TCP 连接。Date 头域指出服务器创建并发送本响应消息的日期和时间，但这并不是对象本身的创建时间或最后修改时间，而是服务器把该对象从其文件系统中取出，插入响应消息中发送出去的时间。Server 头域指出本消息是由 UNIX 服务器上的 Apache 软件所生成的，它与 HTTP 请求消息中的 User-Agent 头域类似。Last-Nodified 头域指出对象本身的创建或最后修改日期或时间。Last-Nodified 头域对于对象的高速缓存至关重要，且不论这种高速缓存是发生在本地客户主机上还是发生在网络高速缓存服务器主机(也就是代理服务器主机)上。Content-Length 头域指明所发送对象的字节数。Content-Type 头域指出包含在附属体中的对象是 HTML 文本。对象的类型是由 Content-Type 头域而不是由文件扩展名正式指出的。

在大多数情况下，除了通用信息头中 Content-Type 之外的所有应答头都是可选的，即 Content-Type 是必需的，它描述的是后面文档的 MIME 类型。虽然大多数应答都包含一个文档，但也有一些不包含，例如：对 HEAD 请求的应答永远不会附带文档。有许多状态代

码实际上用来标识一次失败的请求，这些应答也不包含文档(或只包含一个简短的错误信息说明)。当用户试图通过 HTTP 来访问一台正在运行的 Web 服务器上的内容时，服务器会返回一个表示该请求的状态的数字代码。该状态代码可以指明具体请求是否已成功，还可以说明请求失败的确切原因。可能出现的状态代码及含义如下。

- 1xx:信息响应类，表示接收到请求并且继续处理；
- 2xx:处理成功响应类，表示动作被成功接收、理解和接受；
- 3xx:重定向响应类，为了完成指定的动作，必须接受进一步处理；
- 4xx:客户端错误，客户请求包含语法错误或者是不能正确执行；
- 5xx:服务端错误，服务器不能正确执行一个正确的请求。

响应头域允许服务器传递不能放在状态行的额外附加信息，这些域主要描述服务器的信息和“请求 URI”进一步的信息。响应头域包括 Accept-Ranges、Age、Location、Proxy-Authenticate、Public、Retry-After、Server、Vary、Warning 和 WWW-Authenticate 等。

对响应头域的扩展要求通信双方都支持，如果存在不支持的响应头域，一般将会作为实体头域处理，下面对响应头中一些常用头域进行介绍。

(1) Accept-Ranges 头域

表明服务器是否允许客户使用带有 Range 头域的 GET 方法来取得部分资源。如果服务器支持此功能，则返回“Accept-Ranges:bytes”的消息；否则就会返回“Accept-Ranges:none”。

(2) Location 头域

Location 头域用于重定向接收者到一个新 URI 地址。利用这个功能，可以实现网站的重定向，通常用于网站改变了其域名后，在原域名下设置这种重定向功能，以使用户仍然能看到网站而不至于出现错误提示。

(3) Server 头域

Server 头域包含处理请求的原始服务器的软件信息。此域能包含多个产品标识和注释，产品标识一般按照重要性排序。如早期的 IIS 服务器通常会返回“Microsoft-IIS/5.0”，但为了安全，更高版本的服务器软件则不返回某些敏感信息。

(4) WWW-Authenticate 头域

当服务器上设定了对特定资源的访问权限后，用户必须通过认证才能访问这些资源。当用户访问这些资源时，服务器返回状态码为“401”(Unauthorized)，同时在消息中包含这个头域。客户端浏览器在收到这种响应信息后就会弹出一个要求用户输入用户名和密码的对话框，当用户提供的身份确认后才可以访问这些特定资源。

(5) Proxy-Authenticate 头域

它与 WWW-Authenticate 头域类似，它是为代理服务器对用户的身份认证而设置的。

注意：

HTTP 规范(尤其是 HTTP 1.1)定义了更多可以由浏览器、Web 服务器和网络缓冲服务器插入的头域。此处讨论的 HTTP 请求消息和响应消息中的头域仅是其中很小的一部分，HTTP 规范中定义了更多可用的头部，读者可以查阅相关的 RFC 文档进行更详细的了解。

如何才能查看真实的 HTTP 应答消息呢？其实这非常简单，只需要使用 nc 工具连接到任何一台 Web 服务器，然后输入一行请求消息，用来请求位于该服务器上的某个对象。其中的nc/netcat是一个常用的网络工具，它可以方便在主机之间建立TCP连接并发送命令。例如，可以输入以下指令：

```
nc www.njupt.edu.cn 80
GET /index.shtml HTTP/1.0
```

在输入第二行命令后，连续按两次 Enter 键，就开启了一个到该主机的端口 80 的 TCP 连接，第二条命令就发送了一个 HTTP GET 命令。这样就可以看到服务器的响应消息并附带了该主页的基本 HTML 文件。如果只希望看到 HTTP 消息行而不接收 HTML 文件对象，则可以将上面的 GET 换成 HEAD。

注意：

HTTP 的不同版本可能在上述交互过程中使用不同的命令规范，读者可以查阅相关版本的说明文档进行更详细的了解。

3.1.4　HTTP 应用开发方法

HTTP 协议作为 Web 在技术上的一个重要组成部分，为 Web 的发展和成功奠定了重要的基础。HTTP 提供了客户和服务器进行交互的机制，并对交互过程中所采用的语法和语义进行了规范。从这个意义上说，Web 开发和 HTTP 协议有密切的关系。HTTP 协议从通信的角度贯穿了应用开发的多个层次，包括 HTTP 客户程序、HTTP 服务器程序和服务器端应用程序。



图 3-5　HTTP 应用

1. HTTP 客户程序

HTTP 客户程序实际上就是一种用户代理，可以实现用户和服务器之间的交互，如从服务器下载、向服务器提交信息等。典型的 HTTP 客户程序包括以下几种。

- Web 浏览器：实现 HTML 文档的浏览、下载等功能。常用的有微软的 IE、Netscape Navigator、Firefox、Opera 等。
- 文件下载程序：采用断点续传、多线程等技术为用户提供从服务器上高速下载的功能。常用的软件包括 NetAnts、FlashGet、BitTorrent 等。
- Web 机器人：出于信息检索、资源发现等，而对 Web 进行遍历的程序，通常的搜索引擎就使用了这种方法。

为了说明 HTTP 客户程序的基本结构，这里以某种 Web 浏览器为例，图 3-6 显示了该

Web 浏览器的内部结构。

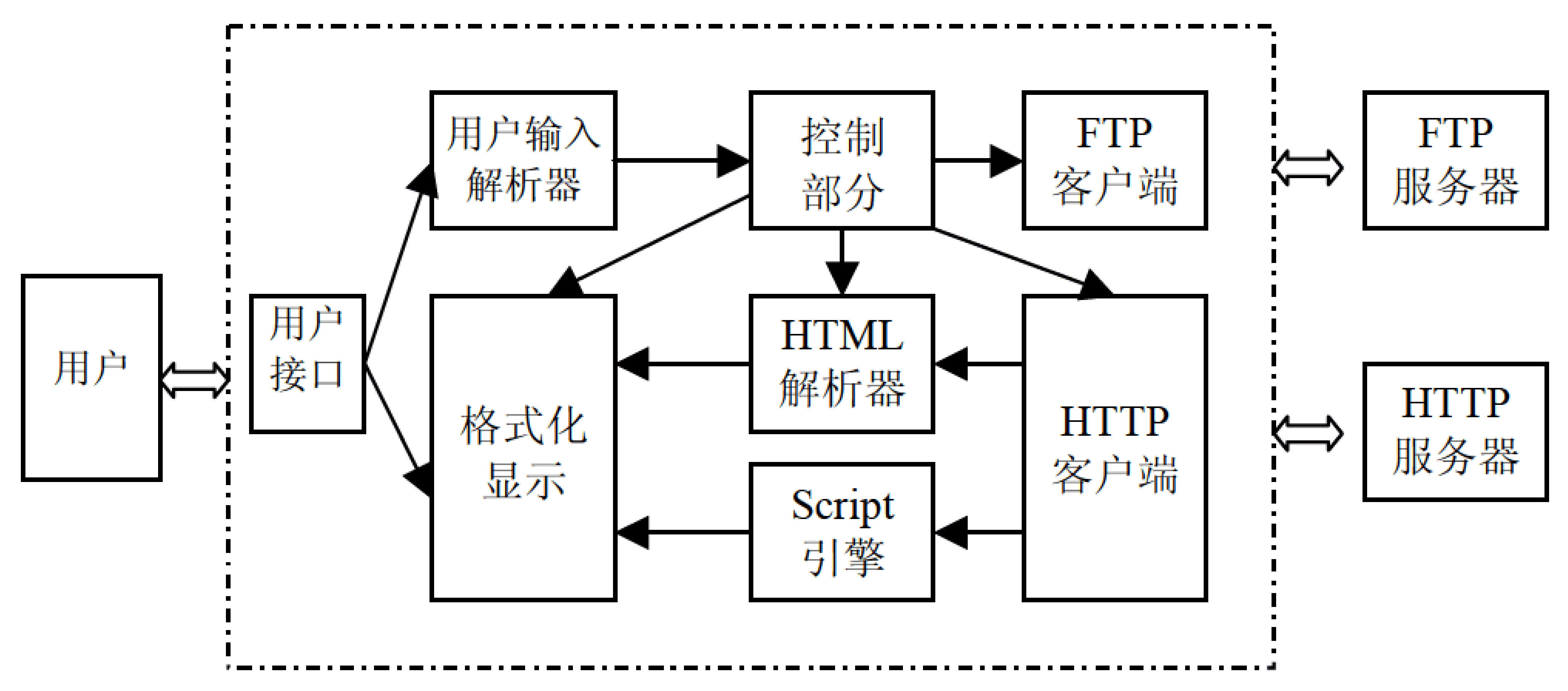


图 3-6　Web 浏览器的基本结构

图 3-6 中主要包括了用户接口、用户输入解析器、控制部分、HTML 解析器、Script 引擎、格式化显示、HTTP 客户、FTP 客户端等，以下进行简要说明。

- 用户接口：负责接收用户的输入，并将格式化显示后生成的结果显示给用户。
- 用户输入解析器：解析用户的输入，将解析结果传送给 Web 浏览器的控制部分。
- 控制部分：一个 Web 浏览器的核心，协调和控制各组成部分之间的协调运行。
- HTML 解析器：如果待显示的内容为 HTML 格式的，那么控制器将文档送至 HTML 解析器，解析器按 HTML 语言的规范进行解析，解析后将结果送到格式化显示模块。本章稍后将对 HTML 规范进行详细讨论。
- Script 引擎：接收到的文档如果包含 Script(脚本)，则由 Script 引擎按照相关标准进行执行，本书将在后继的相关章节中进行详细讨论。
- 格式化显示：将待显示的内容以所要求的显示格式输出到界面上供用户查看。
- HTTP 客户：所有用户使用浏览器发出的 HTTP 请求将通过本模块提交至相应的服务器，并等待 HTTP 服务器的返回信息以便进行接收和解析。
- FTP 客户：通过浏览器软件也可以使用其他的协议所提供的非 Web 服务，因此需要相关的客户模块，除了这里表示的 FTP 客户外，浏览器往往还支持其他的一些协议，如新闻组、电子邮件等。

2. HTTP 服务器程序

即 HTTP 服务器或 HTTP 代理，它们都可以接收 HTTP 请求，并提供相关的服务。

HTTP 服务器程序的作用是为 HTTP 客户提供服务。其中起关键作用的是 HTTP 协议，HTTP 服务器程序和 HTTP 客户都按照这个协议来完成交互的过程，包括客户与服务器之间的连接、客户请求消息的解析、客户所要求的处理、响应消息的语法格式等。因此一个 HTTP 服务器必须实现 HTTP 协议的内容，如果要开发 HTTP 服务器程序，必须较为透彻地理解 HTTP 协议中关于客户和服务器交互的机制，此外还需要掌握各种消息的语法和语义规范。在这个功能的基础上，HTTP 服务器程序还需要具备传递功能，将客户所提交的诸如 CGI 和 ASP 等格式的请求传递给 CGI 程序或 ASP 脚本服务器端程序，接收这些服务器端应用程序处理后所返回的结果，这些扩展功能依赖于服务器与服务器端应用程序的接

口规范，如 CGI、ISAPI 等。该功能的实现过程如图 3-7 所示。

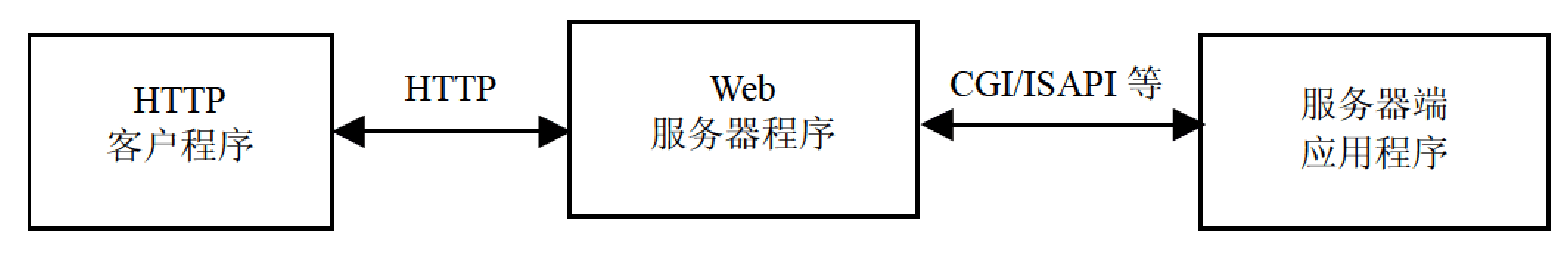


图 3-7　HTTP 服务器的功能扩展

HTTP 代理是 HTTP 和 HTTP 服务器之间的中介，通常我们所使用的代理服务器就是属于这种应用。首先 HTTP 代理服务程序充当 Web 服务器，接收客户的请求；然后经过必要的中间处理过程，如身份验证、日志、安全、过滤、缓存等；最后将请求发送给目标 Web 服务器；反之也是一样，其工作原理如图 3-8 所示。

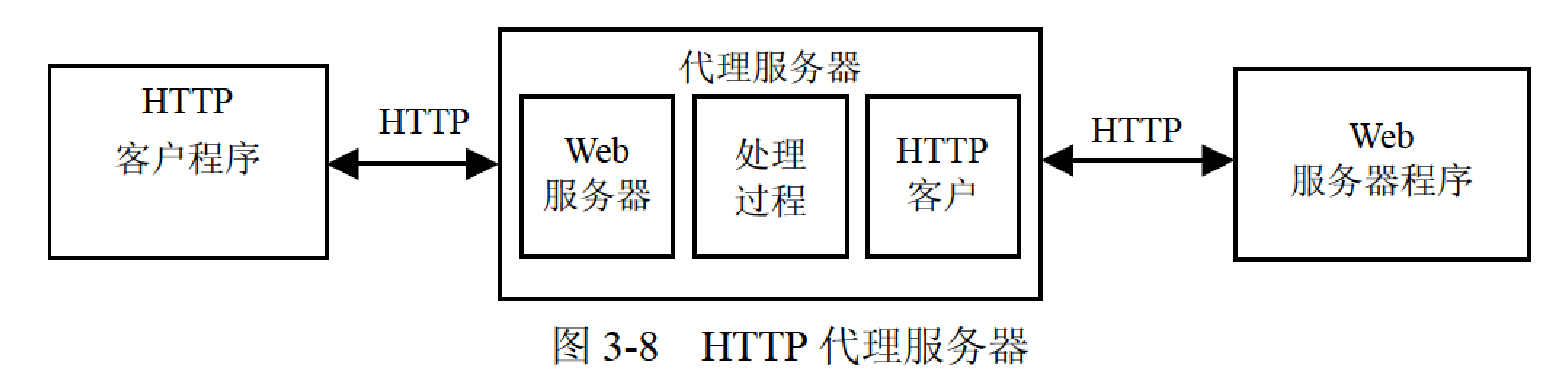


图 3-8　HTTP 代理服务器

3. 服务器端应用程序

服务器端应用程序的作用是：根据用户提交的信息实时生成响应。它必须符合特定的接口规范，如 CGI、ISAPI/NSAPI、Servlet 或 JSP 等，以便接收 Web 服务器传递过来的参数，将处理好的结果返回给 Web 服务器，最后由 Web 服务器将信息利用 HTTP 协议返回客户端。服务器端应用程序与 Web 服务器之间的交互过程如图 3-9 所示。

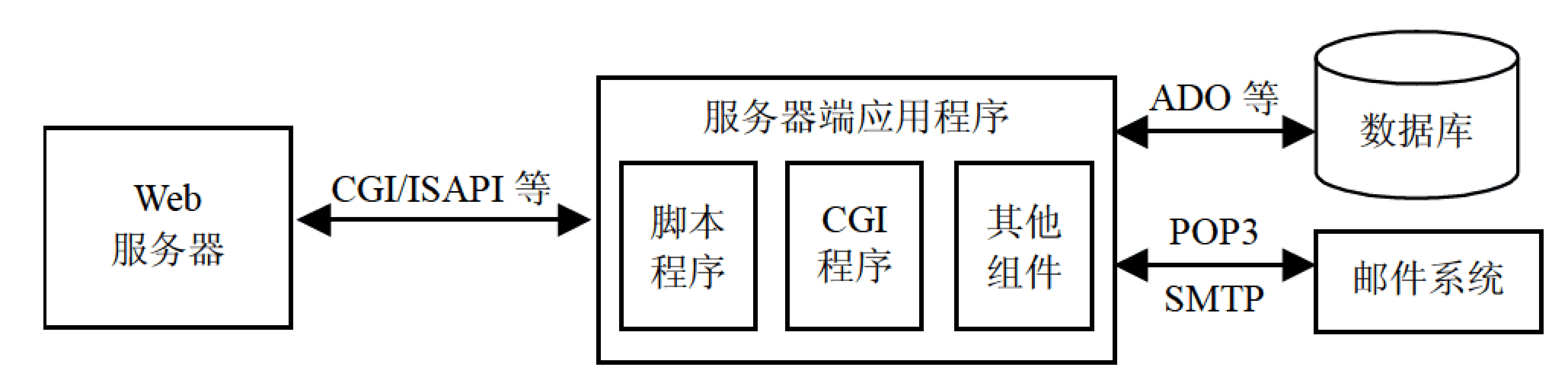


图 3-9　服务器端应用程序与 Web 服务器之间的交互

这里需要说明的是：除了实现基本的功能外，HTTP 服务器还需要具备其他一些功能，如根据需要访问数据库、访问 WebService、与邮件系统交互等。这些功能使用了其他的规范或协议，如 ADO、WSDL、UDDI、SOAP、POP3 以及 SMTP 等。对于部分不属于本书的内容，请读者参考其他有关书籍。

3.1.5　HTTP 应用的开发

前面介绍的 HTTP 客户程序、HTTP 服务器程序、服务器端应用程序三类应用中的前两类是与 HTTP 协议密切相关的。应用程序通常建立在系统的应用编程接口(API)之上，HTTP 应用程序的开发也不例外。根据不同的 API 抽象层次，可将 HTTP 应用程序的开发分为两种：一种使用了网络层的编程接口，如 Socket API 等；另一种使用了应用层的 API，如 Internet Transfer 控件等。

1. Socket 应用编程接口

Socket 接口是基于 TCP/IP 的通信编程接口，它首先是在 UNIX 系统上提出的，且最初只能用于 UNIX 系统。随着计算机应用越来越广泛，Socket 在 UNIX 的成功应用使得将 Socket 移植到 DOS 和 Windows 平台成为一件有意义的工作。20 世纪 90 年代初，连同微软在内的多家公司共同制定了一套标准，即 Windows Socket 规范，将 Socket 机制引入了 Windows，先后推出了 WINSOCK 1.0、WINSOCK 1.1、WINSOCK 2.0，由于 Windows 操作系统与 UNIX 系统任务调度方式的区别，WINSOCK 除了可以兼容 UNIX 和 Socket 编程接口外，又把它加以扩展以适合 Windows 文件驱动特性。由于 Windows 编程方法相对复杂，为此，现在提供的编程语言中，都将相关功能封装到一些类中，通过这些类来编写基于 Windows 系统下的网络通信程序。

利用基于 TCP/IP 的 Socket API 编写程序，其目的是在 TCP/IP 所组建网络的不同计算机之间建立通信连接。为建立该连接，开发人员只要提供一些基本的连接信息，其余由操作系统内核来完成，下面讨论建立一个完整通信连接开发者需要提供的信息。以机器 A 通过 TCP/IP 与计算机 B 进行网络通信为例，对于计算机 A 来说需要知道如下信息：

- 计算机 B 的 TCP/IP 地址；
- 与计算机 B 中哪一个进程(或软件系统)联系。

这里所需要的两个参数，第 1 个很难理解，但对于第 2 个也许有人存在疑问，因为电子邮件收发、TELNET 等基本的 TCP/IP 网络应用程序，在建立连接时都只要提供 TCP/IP 地址(或者对应域名地址)，根本没有必要提供第 2 个参数，这是为什么呢?其实原因很简单，就是它们使用的是一些标准接口，第 2 个参数是事先确定好的，如 HTTP 为 80。这些应用系统在发出呼叫请求前，自己已经知道该与对方怎样联系，在发送请求前，程序已经自动将第 2 个参数加入请求中。如果开发者也要开发这样的系统，就需要知道这些标准接口；对于那些需要建立专用系统网络连接的，却需要双方协商。

这两个参数，在 Socket 套接字中分别表示为计算机 B 的地址和计算机 B 的通信端口。通过在同一计算机的不同通信软件中定义不同端口地址，来表示计算机 A 是与计算机 B 中哪个应用通信。不管是利用何种协议，完全建立一个网络连接需要 5 个基本信息。它们分别是，双方的地址、约定的通信端口和协议类型。Socket 通信编程接口并不是专门为 TCP/IP 通信提供的，因此套接字通信编程需要在参数中指明通信协议类型。套接字是利用客户/服务器模式来实现通信的，客户端软件和服务器端软件的具体实现也有所不同。

注意：

读者可以在 Windows 系统目录下的“\system32\drivers\etc”子目录下找到一个名为“services”的文件，这个文件中给出了常用端口和对应的协议。由于没有扩展名，双击后不能直接打开，读者可以使用任何文本编辑器打开这个文件后查看。

具体来说，在客户端利用基于 TCP/IP 和 Socket 通信编程的基本步骤如下。

- 声明一个套接字类型的变量，需要在该变量定义中提供本机 IP 地址和通信端口并

指明协议类型，由于在此介绍的是基于 TCP/IP 的套接字通信，因此协议类型应该是 TCP/IP，在编程接口中该类型用 AF-INET 来表示。

- 向对方发出连接请求，连接时编程者需要提供对方 TCP/IP 地址和通信端口，同时 SOCKET 实现程序自动向对方提供本机 TCP/IP 地址和通信端口。
- 如果连接成功，会收到对方的应答信号，这以后的通信就可以通过套接字的相关操作来实现了。

利用 SOCKET 来实现服务器端通信软件的步骤如下：

- 同客户端程序第 1 个步骤；
- 服务器端通信软件进入等待客户端连接的状态，如果收到连接，则从对方连接请求中获取对方的 IP 地址和通信端口，并向对方发送连接成功的应答信号。

客户端与服务器端通过 SOCKET 通信都需要知道 5 个基本信息，不同的是客户端软件开发者需要向编程接口提供全部 5 个参数，而服务器端软件开发者只需要提供 3 个。

需要注意的是：在一般的通信过程中，客户端和服务器端所提供的本地端口地址可以相同，也可以不同，但客户端口地址可以动态分配，而服务器端端口地址必须固定，否则连接就不能建立(P2P 的通信过程是个例外，此处暂不讨论)。

2. 应用层的 API

为了快速开发网络应用程序，实际上也可以使用一些现成的函数库或者控件。某些函数库或控件已经封装了 HTTP 客户的功能，通过所定义的接口提供给开发者使用。这种开发不需要涉及 HTTP 协议本身以及网络层上的实现细节，提高了开发的效率。

例如，上面提到的 Internet Transfer 控件，就同时支持超文本传输协议(HTTP)和文件传输协议(FTP)，它具有如下的属性：AccessType、Document、hInternet、Index、Name、Object、Parent、Password、Protocol、Proxy、RemoteHost、RemotePort、RequestTimeout、ResponseCode、ResponseInfo、StillExecuting、Tag、URL 和 UserName，并可以调用如表 3-1 所示的方法。

表 3-1　Internet Transfer 控件的方法

方　法　名	用　　途
OpenURL	下载指定网址的 HTTP 网页
Execute	执行对远程服务器的请求，只能发送对特定的协议有效的请求
GetChunk	检索缓冲区的内容
GetHeader	检索 HTTP 文件的头信息
Cancel	取消当前请求，并关闭当前创建的所有连接

通过表 3-1 可以看出，该控件将常用功能进行了封装，通过属性、事件和函数的参数来调用。严格的说，这种开发是一种面向组件的编程，较为方便；但因为受到接口的限制，某些特殊功能是实现不了的。详细的开发方法请参考相关的手册和说明，这里不再赘述。

3.1.6　安全超文本转移协议(HTTPS)及安全套接层(SSL)

1. 安全超文本转移协议(HTTPS)

安全超文本转移协议(Secure Hypertext Transfer Protocol，简称 S-HTTP)是一种结合 HTTP 而设计的消息的安全通信协议。S-HTTP 协议为 HTTP 客户机和服务器提供了多种安全机制，这些安全服务选项是适用于 Web 上各类用户的。还为客户机和服务器提供了对称能力(及时处理请求和恢复及两者的参数选择)同时维持 HTTP 的通信模型和实施特征。

S-HTTP 不需要客户方的公用密钥证明，但它支持对称密钥的操作模式。这意味着在没有要求用户个人建立公用密钥的情况下，会自发地发生私人交易。它支持端对端安全传输，客户机可能首先启动安全传输(使用报头的信息)，用来支持加密技术。

在语法上，S-HTTP 报文与 HTTP 相同，由请求行或状态行组成，后面是信息头和主体。请求报文的格式由请求行、通用信息头、请求头、实体头、信息主体组成。响应报文由响应行、通用信息头、响应头、实体头、信息主体组成。

可以有两种方法来建立连接：HTTPSURI 方案(RFC 2818)和 HTTP 1.1 请求头(由 RFC2817 引入)。由于浏览器对后者几乎没有任何支持，因此 HTTPS URI 方案仍是建立安全超文本协议连接的主要手段。安全超文本连接协议使用 https://代替 http://。

2. 安全套接层(SSL)

安全套接层(Secure Sockets Layer，SSL)，及其继任者传输层安全(Transport Layer Security，TLS)是为网络通信提供安全及数据完整性的一种安全协议。TLS 与 SSL 在传输层对网络连接进行加密。

安全套接层协议能使用户/服务器应用之间的通信不被攻击者窃听，并且始终对服务器进行认证，还可选择对用户进行认证。SSL 协议要求建立在可靠的传输层协议(TCP)之上。SSL 协议的优势在于它是与应用层协议独立无关的，高层的应用层协议(如 HTTP、FTP、TELNET 等)能透明地建立于 SSL 协议之上。SSL 协议在应用层协议通信之前就已经完成加密算法、通信密钥的协商及服务器认证工作。在此之后应用层协议所传送的数据都会被加密，从而保证通信的私密性。

SSL 协议可分为两层：SSL 记录协议(SSL Record Protocol)，它建立在可靠的传输协议(如 TCP)之上，为高层协议提供数据封装、压缩、加密等基本功能的支持。SSL 握手协议(SSL Handshake Protocol)，它建立在 SSL 记录协议之上，用于在实际的数据传输开始前，通信双方进行身份认证、协商加密算法、交换加密密钥等。如果要启用 SSL 通道，那么需要使用 SSL 证书来启用 HTTPS 协议。

SSL (Secure Socket Layer)为 Netscape 所研发，目前一般通用之规格为 40 bit 安全标准，美国则已推出 128 bit 更高安全标准，但限制出境，当前版本为 3.0。目前几乎所有浏览器都能支持 SSL。

3.2　HTML 基础

HTTP 协议规定了 Web 客户端和服务器之间进行交互的通信协议，它解决的是通过请求和响应来传递信息资源的问题，但对于消息中资源实体的格式、类型等并没有作出规定。这个任务就是交由 HTML 语言来完成的。

本部分将介绍数据表示规范 HTML，包括 HTML 的基本语法和语义。

3.2.1　HTML 简介

1. 超文本

传统的知识资源如图书、文章、文件等，所采用的多是线性的顺序结构，而真实世界的知识资源实际上是以非线性网状结构来组织的，如人脑中存储的知识。

早在 15 世纪 30 年代，美国著名科学家 V.Bush 开始担心信息量增长会使专家无法跟踪学科的发展，他还指出了现有共享与表现信息的方法少而不敷应用及传统顺序检索方法的缺点，并提出了一种叫作 Memex 的设想，其检索方法试图采用联想机制，这便是超文本的雏形。

超文本(Hypertxt)这个术语是美国的 TedNelson 于 20 世纪 60 年代提出来的，他还设想了一个名为 Xanadu 的系统。Nelson 认为，“任何事物都有很深的联系”，超文本作为一种文字媒介能以非线性存储任何东西。

Nelson 提出超文本后，对超文本的研究得到了许多人的重视，也取得了可喜成果。一些基于超文本概念和技术的系统相继研究成功并投入使用。到 20 世纪 80 年代，超文本研究发生了质的飞跃，超文本技术得到越来越广泛的应用。

超文本是一种信息管理方式，它的本质含义是非线性的文档组织形式；是采用了符合人脑思维模式的联想机制对庞大的信息资源进行索引的一种非线性结构。实际上超文本文档都是静态的文件，这个文件里面包含了某种指令代码，这些指令代码并非通常的程序语言，而只是一种页面的排版规则，定义资源显示位置的结构标记语言。超文本文档由节点(Node)和链(Link)组成，允许用户从一个主题跳转到另一个主题，从一个页面跳转到另一个页面。超文本可以被理解为“超级链接＋文本”的文本组织形式，图 3-10 展示了以超文本方式组织的内容所构成的网状关系。正是因为超文本具有网状的逻辑结构，它符合人类思维的联想型方式，也许这才是 Web 得以流行的深层原因。

2. 超媒体

20 世纪 80 年代以后，超文本得到了进一步的发展。由于超文本技术对信息的管理是基于信息块的，基于此，超文本就不仅可以处理文本信息，还可以处理图形、图像、声音、动画、视频乃至它们的组合，这样自然而然形成了超媒体的概念。即超媒体＝超文本＋多媒体。

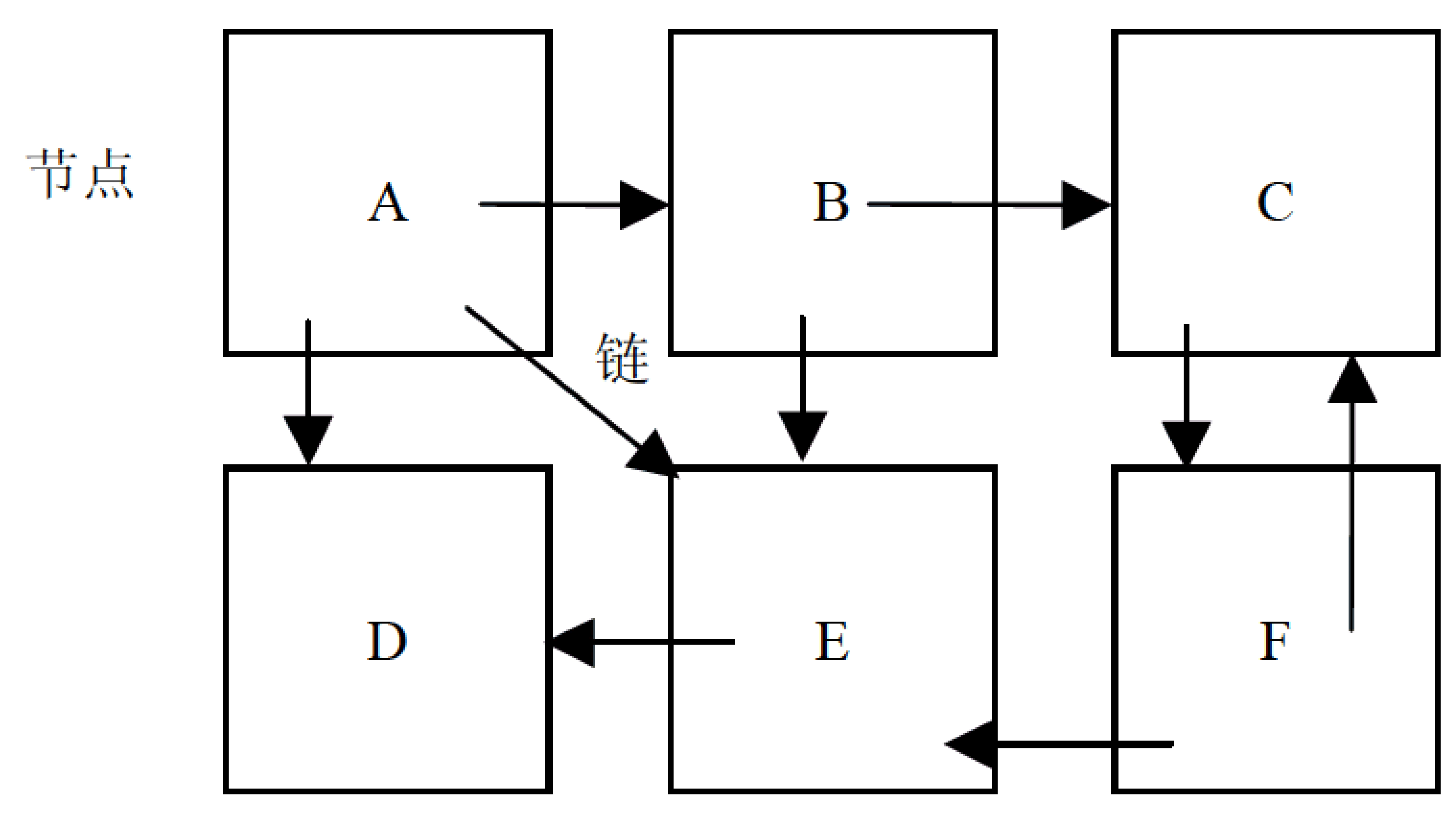


图 3-10　超文本组织的网状关系

Apple 公司在 Macintosh 平台上推出的 Hypercard 软件支持层次的网状结构并开始使用图形用户界面(GUI)，它使得超媒体不再停留在概念和实验的阶段。其后微软的 Windows 系统亦采用了超文本式的帮助功能，可以让使用者在寻求帮助时更为便捷。微软在推出 Windows 3.0 时，曾附送另一个超文本系统 Toolbook，但不及 Hypercard 普遍。

但也有人认为没有必要为一个特殊的超文本保留一个专门的术语，多媒体超文本也是超文本。但无论如何，多媒体的引入，使超文本界面更加生动，信息的表达和交互方式更加丰富，从而使超媒体技术具有了更广阔的潜力和魅力。

Internet 的迅速发展及 Web 的出现，使超文本的应用更加广泛。超文本正在朝网络型、智能化、超媒体方向发展，超文本和超媒体的应用也越来越广泛。虽然传统超文本的功能已经非常强大了，但它具有一个重大的缺陷：仅能将同一台计算机上的文件以超文本的方式进行链接。

3. HTML

为了理解 HTML，我们首先给出 W3C 对于 HTML 的定义：HTML is the lingua franca for publishing hypertext on the World Wide Web. It is a non-proprietary format based upon SGML, and can be created and processed by a wide range of tools, from simple plain text editors – you type it in from scratch – to sophisticated WYSISYG authoring tools. HTML uses tags such as <h1> and </h1> to structure text into headings paragraphs, lists, hypertext links etc.

上面的文字告诉我们：HTML 是一种国际化标准语言，它用于在 Web 上发布超文本信息，是一种基于 SGML、公开的资源描述格式。创作者可以使用任意的工具来创建和处理 HTML 文档，包括简单的文本编辑器(一个个字符从零开始录入)以至复杂的具有所见即所得功能的编辑工具。HTML 使用诸如<h1>和</h1>的标签将文本组织成结构化的形式，例如，标题、段落、列表、超链接等。

上面提到的 W3C(World Wide Web Consortium，即：万维网联盟/环球网协会，网址是 http://www.w3.org)是 1994 年 10 月成立的一个国际化组织，负责万维网的标准制订和技术开发。W3C 通过公开出版(非商业性的)万维网语言和协议标准，致力于维护万维网的统一和良性发展。

根据上面的定义，可以发现 HTML 的几个重要的特征。

- 它是 Web 发布信息的方式：HTML 是在 Web 上发布信息所使用的语言，它能将信息有效地组织起来，方便用户浏览。
- 这种规范是非私有的：属于一种公有的国际规范，它不属于任何个人或者组织。这种开放性的特点使得 Web 能真正做到国际化，全球范围的信息共享得以实现。具体来说，标准化的工作是由 W3C 的 HTML 工作组负责的。
- 它是描述文档的标记语言：HTML 语言作为一种标记性的语言，是由一些特定符号和语法组成的，用标签(tag)来说明文档的内容与格式。HTML 的文档都是 ASCII 文本文件，普通的字符编辑器即可创建。采用专用的 HTML 编辑工具，还能够实现自动检查 HTML 文档中的语法错误并协助改正。
- 支持超链接：可以在文档中建立与其他文档的联系，构成网状的结构。
- 具有平台无关性：由于 HTML 语言是标记性语言，它在浏览器中无须编译，是解释执行的；因此 HTML 语言编写的文档可以在各种浏览器中进行浏览。这决定了 HTML 文档是独立于平台的，具有跨平台性。
- 具有可扩展性：HTML 语言的广泛应用导致了更多的需求，对此可以采用增加标签等方式来满足，HTML 采取子类元素的方式，使系统在一定范围内进行扩展。
- 基于 SGML：标准通用标记语言 SGML(Standard Generalized Markup Language)是 ISO 于 1986 年制定的，主要用于生成独立于平台的应用和文档，在文档中定义了格式、索引、链接等信息。但由于过于复杂，1996 年经裁减和简化推出了可扩展标记语言 XML(eXtensible Markup Language)。HTML 也遵循了类似的思路，但由于目标不同，它与 XML 存在差异，图 3-11 表示了 HTML 与相关语言标准的关系。

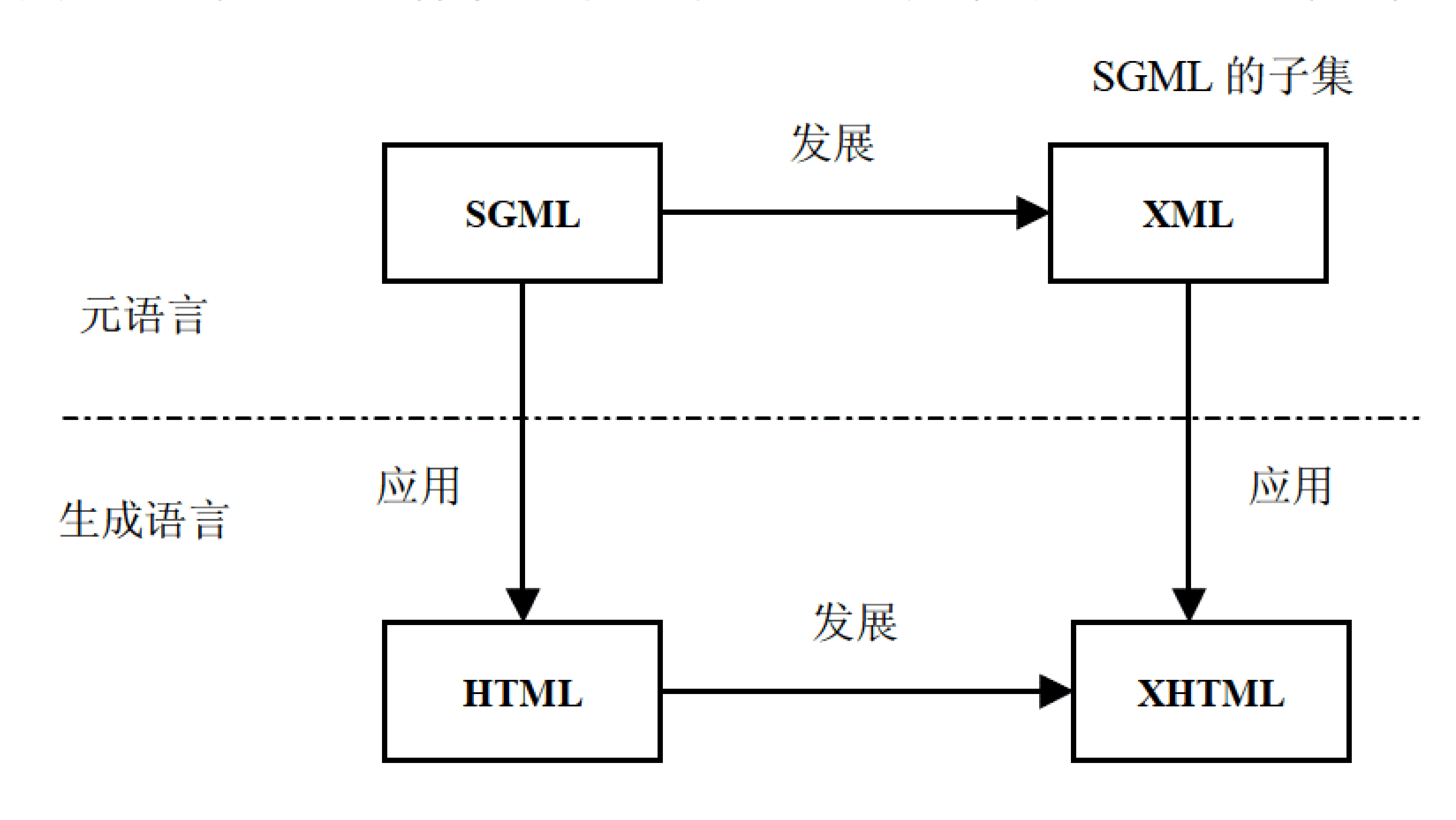


图 3-11　HTML 与相关语言标准的关系

4. HTML 的发展历史

万维网之父蒂姆 • 贝纳斯 • 李(Tim Berners-Lee)是英国人，1976 年毕业于牛津大学的皇后学院。为了弥补自己健忘的缺陷，1980 年他编写了一个可以把易忘的东西联系在一起的软件 Enquire(询问)。后来在 CERN(Conseil Europeen pour la Recherche Nucleaire，欧洲粒子物理研究所)工作期间，他在 Enquire 的基础上，于 1989 年提出全球超文本计划(global hypertext project)，后来取名为万维网(WWW 全称为 World Wide Web，又称环球网)。

1990 年 Berners-Lee 写出了第一个万维网服务器(HTTP)和第一个客户机软件(HTML 浏

览器)。1990 年 12 月，万维网开始在 CERN 内部推出，1991 年夏天开始在互联网上流行。1994 年 Berners-Lee 加入麻省理工学院(MIT)计算机科学系的人工智能实验室，并牵头成立了万维网联盟(World Wide Web Consortium，简称 W3C)，任该联盟的总监(Director，主任)至今。

网景(Netscape)公司创始人马克•安德森(Marc Anderressen)是美国人，他在伊利诺伊(Illinois)大学读研期间，到该校内的 NCSA(National Center for Supercomputing Applications 美国超级计算应用中心)打工。1993 年 3 月，他与好友埃里克 • 比纳(Eric Bina)合作开发出支持内嵌图像的浏览器马赛克(Mosaic 镶嵌图案)0.9 版，1993 年 11 月又推出马赛克 1.0 版，并在网上迅速扩散。

1994 年 4 月安德森与 SGI 公司的创始人吉姆•克拉克(Jim Clark)共同创办了网景公司，安德森等人又重写了马赛克的代码，并于 1994 年 10 月推出了 Navigator 浏览器，后来改名为 Netscape 浏览器。1995 年 Netscape 公司的 Brendan Eich 发明了 JavaScript，为 HTML 浏览器提供了脚本功能和动态网页能力。

1995 年微软公司从伊利诺伊大学购得马赛克技术，并在此基础上开发出 IE(Internet Explorer 互联网探索者)浏览器，并推出了与 JavaScript 功能类似的 JScript 和 VBScript。

网景公司后来被微软公司随 Windows 操作系统免费赠送的 IE 浏览器挤垮，于 1998 年 11 月被 AOL 公司收购，安德森当时成为该公司的 CTO(首席技术官)。1999 年 9 月他离开了 AOL，并创立了一家基于服务的 Web 主机公司 Loudcloud，2002 年更名为 Opsware。

目前的几大浏览器，如微软的 IE、Mozilla 的 Firefox(火狐)等都是基于 Mosaic 的。其中，Mozilla 是 2005 年 8 月 3 日成立的一家非盈利公司，由 AOL 的 Netscape 分部于 2003 年 7 月成立的 Mozilla 基金资助。

总之，Tim Berners-Lee 发明了具有超文本能力的万维网(HTML/HTTP)，Marc Anderressen 等人将图像、多媒体、交互动态等功能带入万维网，促进了互联网的发展和普及，成就了今天万维网的繁荣兴旺。

从 1990 年 HTML 的应用期开始，HTML 得到了广泛的应用。为了满足在实际应用中出现的新的需求，HTML 规范得到了扩充，增加了不少的新特性和新功能。这些新的扩展一旦得到广泛的认同，就被融入新版本的 HTML 规范中。以下就是 HTML 发展的重要里程碑。

(1) 开创期(1932~1965 年)

- 1932 年美国总统罗斯福的科学顾问 V. Bush(布什)提出一种非线性文本结构——Memex (memory extender，记忆延伸器)，1945 年才公开发表，当时曾引起广泛注意。布什是超文本的先驱，被公认为是超文本的创始人。
- 1965 年美国的 Ted Nelson 造“hypertext”(超文本)一词来表示信息以复杂形式相连，1974 年出版有影响的书 Computer lib / dream machines(计算机库/梦想机器)，20 世纪 70 年代末开始开发联机全球图书馆——Xanadu(世外桃源)数字图书馆/数字地球，至今未成。

(2) 研究期(1965~1985 年)

- 1967 年美国布朗(Braun)大学的 A. Van Dam 研制成首个超文本系统——The Hypertext Editing System(超文本编辑系统)，1968 年又推出文件检索与编辑系统 FRESS(File Retrieval and Editing System)。
- 1975 年美国卡内基—梅隆大学(CMU)推出知识库管理系统 KMS(Knowledge Management System)(原称为 ZOG)。
- 1978 年美国麻省理工学院(MIT)的 A. Lippman 制成首个超媒体视频盘片系统——Aspen Movie Map(白杨城地图集)。

(3) 成熟期(1985~1990 年)

- 1985 年苹果(Apple)公司的 Janet Walker 推出首个商品化超文本系统——Symbolics Document Examiner(符号文献检测器)，它运行于 Macintosh 机上。
- 1985年美国布朗(Braun)大学的 N. Meyrow 推出交互媒体(Intermedia)，也在 Macintosh 机上运行。
- 1986 年办公工作站有限公司(OWL)推出第一个广泛应用的超文本系统——Guid(指南)。
- 1987 年施乐(Xerox)公司的 Halasz 推出 Notecards(笔记卡)系统。
- 1987 年苹果(Apple)公司的 Bill Atkinson 在 Mac 机中引入 HyperCard(超卡)系统。
- 1987 年 11 月召开 ACM(Association for Computing Machinery，[美国]计算机协会)超文本专题讨论会。
- 1989年6月召开首届超文本大会，出版第一本超媒体专业杂志 Hypermedia(超媒体)。

(4) 应用期(1990 年至今)

- 1990 年英国科学家 Tim Berners-Lee 发明万维网，设计出 HTTP 和 HTML。
- 1991 年美国的 Asymtrix 公司推出基于超文本的多媒体著作工具——ToolBook。
- 1993 年美国人 Marc Anderressen 和 Eric Bina 合作开发出支持内嵌图像的浏览器马赛克(Mosaic，镶嵌图案)。
- 1998 年 W3C(World Wide Web Consortium，万维网联盟)发布 XML 1.0。
- 2000 年 W3C 推出基于 XML 的 XHTML 1.0。

5. HTML 的版本

HTML 的版本有：

- 0.9——1990，基本超文本(Berners-Lee & Connolly)。
- 1.0——1992.1。内嵌图、文字格式(CERN)。
- 2.0——1995.11。表单(IETF：RFC 1866)。
- 3.0——1996。数字、表格、控件；未公开、未标准化(W3C)。
- 3.2——REC: 1997.1.14。规范对 Applet、脚本和颜色的支持(W3C：REC 标准)。
- 4.0——REC: 1997.12.18 (1998.4.24 推出修订版)。提倡结构与外观的分离，支持 CSS(Cascading Style Sheets，层叠样式表单)，提高了对(动态)页面的控制能力、改

进外观和功能、支持多语言文档(W3C：REC 标准)。

- 4.01——REC: 1999.12.24，对 4.0 的修正与补充，如更新样式表单、添加内容短表、修正文档脚本等(W3C：REC 标准)。
- 5.0——自 1999 年 12 月发布的 HTML 4.01 后，后继的 HTML 5 和其他标准被束之高阁，为了推动 Web 标准化运动的发展，一些公司联合起来，成立了一个叫作 Web Hypertext Application Technology Working Group (Web 超文本应用技术工作组 -WHATWG) 的组织。WHATWG 致力于 Web 表单和应用程序，而 W3C 专注于 XHTML 2.0。在 2006 年，双方决定进行合作，来创建一个新版本的 HTML。HTML 5 的第一份正式草案已于 2008 年 1 月 22 日公布；2013 年 5 月 6 日， HTML 5.1 正式草案公布。

因为 W3C 想用 XHTML 来代替 HTML，因此不断推出了这个系列的不同版本，主要的版本有：

- 1.0——可扩展超文本标记语言，基于 XML 的 HTML，HTML4 的 XML 重写；
- 1.0 (第二版)——REC: 2002.8.1；
- 1.1——基于模块的 XHTML；
- 2.0——针对丰富、移动的基于 Web 的应用，不向后兼容。

从 HTML 到 XHTML 的版本演化，可参见图 3-12。

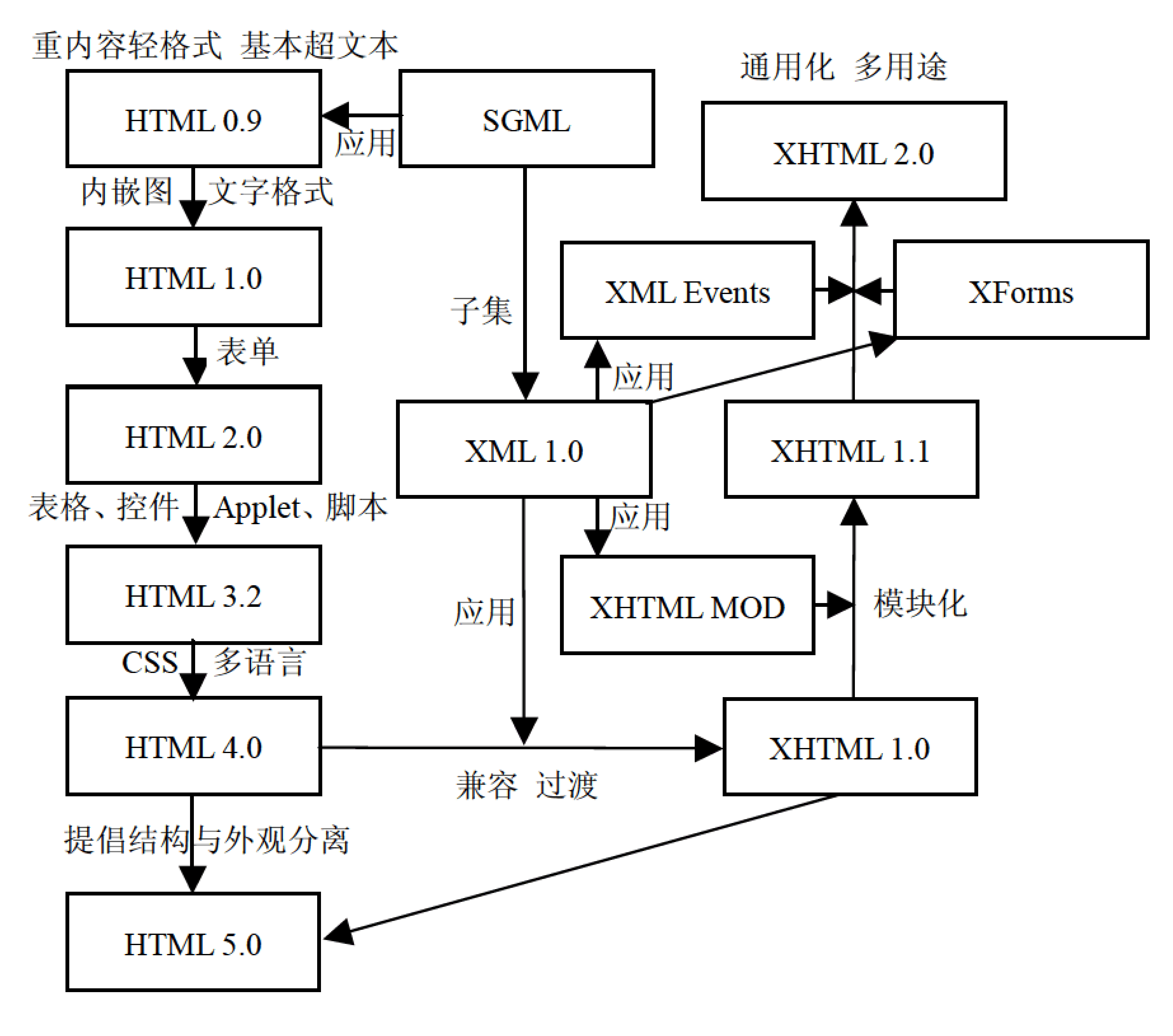


图 3-12　从 HTML 到 XHTML 的版本演化

3.2.2　HTML 标记语法及文档结构

创建一个 HTML 文件，只需要两个工具，一个是 HTML 编辑器，一个是 Web 浏览器。HTML 编辑器是用于生成、修改和保存 HTML 文档；Web 浏览器用于打开并显示网页，

所看到的网页是由浏览器解释 HTML 标记形成的可视化图像、文字的集合。

HTML 只是一个纯文本文件，由“显示内容”及“控制语句”两部分组成。其中的控制语句描述了显示内容以何种形式在浏览器中显示，为了与显示内容区分，控制语句是以标签的形式出现的。

1. HTML 的标签与标签的属性

HTML 的标签根据其书写形式可分为单标签和双标签。

(1) 单标签

某些标签称为“单标签”，这种标签只需单独使用就能完整地表达意思，其语法为：

```
<标签名称>
```

最常用的单标签有：
表示换行，<HR>表示一条水平线等。

(2) 双标签

另一类标签称为“双标签”，由“首标签”和“尾标签”两部分构成，必须成对使用。首标签告诉浏览器从此处开始执行该标签所表示的功能，而尾标签通知浏览器在这里结束此项功能。首标签前加一个斜杠(/)即成为尾标签。这类标签的语法是：

```
<标签>受标签影响的内容</标签>
```

如果希望对某段文字进行加粗显示，则将此段文字放在<B>..</B>标签中，其语法是：

```
<B>你要加粗的字</B>
```

(3) 标签的属性

许多单标签和双标签的始标签内可以包含一些属性，标签要通过属性来定义所希望的设置参数。其语法是：

```
<标签 属性 1=属性值 属性 2=属性值 …>受影响内容</标签>
```

注意：

并不是所有的标签都有属性；根据需要可以用该标签的所有属性，也可以只定义需要的几个属性，说明时属性之间没有顺序。

作为一般的原则，大多数属性值不用加双引号。但如果参数中包括了空格、“%”和“#”等特殊字符的属性值则必须加双引号。为了形成良好的习惯，提倡对全部属性值均加双引号。例如：

```
<font color="#ff00ff" face="宋体" size="30">字体设置</font>
```

在输入始标签时，一定不要在“<”与标签名之间输入多余的空格，也不能在中文输入法状态下输入这些标签及属性，否则浏览器将不能正确地识别括号中的命令，从而无法正确地显示信息。

(4) 字符引用

浏览器解释 HTML 文件时，是根据“<”与“>”来识别标签的，然后再确定这两个符号中的内容是否为 HTML 文件标签，若是则按其规则解读。因此，网页中的“<”或“>”会被浏览器作为标签解析，不能直接显示出来。如果希望在网页中显示“<”或“>”，则需要将它们作为特殊字符，这种用法被称为字符引用。字符引用以“&”开始，以“;”结束。它有两种形式：数值字符引用和字符实体引用。

① 数值字符引用。字符的数值表示通常为“数值字符引用”(Numeric Character Reference)。它通过给出字符在字符集中的代码(十进制或十六进制)来表示字符。比如十进制字符引用“<”、十六进制字符引用“<”和“&#X3c”均表示了字符“<”。

② 字符实体引用。字符的符号表示通常为“字符实体引用”(Character Entity Reference)。这种引用方式是通过字符的名称(助记符，大小写敏感)来表示字符的。例如：“>”表示了字符“>”，其中的 gt(greater than)是大于符号的助记符。这种方式为 HTML 文档的作者提供了一种直观的表示字符的方法，相比于上面介绍的数值字符引用更加方便和容易记忆。其他常用的字符实体引用如表 3-2 所示。

表 3-2　常用的字符实体引用

字符实体引用	显 示 结 果	描　　述
&		特殊字符的开始
;		特殊字符的结束
<	<	小于号或显示标识
>	>	大于号或显示标识
&	&	可用于显示其他特殊字符
"	"	引号
®	®	已注册
©	©	版权
™	™	商标
		不断行的空白

(5) 注释

像很多计算机语言一样，HTML 文件也提供注释功能。浏览器会忽略此标签中的文字(可以是很多行)而不做任何处理，也不进行显示。一般来说，其使用目的是为文中不同部分加上说明，方便作者和阅读 HTML 源文件的人日后理解和修改，注释标签的格式为：

```
<!--  注释内容  -->
```

需要注意的是：注释并不局限于一行，长度不受限制，且结束置标符“<!--”不必与开始置标符“-->”在同一行上。

2. HTML 文件的总体结构

一个完整的 HTML 文件包括标题、段落、列表、表格以及各种嵌入对象，这些对象统称为 HTML 元素。在 HTML 中使用标签来分割并描述这些元素。实际上可以说，HTML

文件就是由各种 HTML 元素和标签组成的。一个 HTML 文件的基本结构如下：

```
<HTML>          文件开始标签
<HEAD>          文件头开始的标签
…               文件头的内容
</HEAD>  文件头结束的标签
<BODY>          文件主体开始的标签
…               文件主体的内容
</BODY>  文件主体结束的标签
</HTML>  文件结束标签
```

从上面的代码结构可以看出，在 HTML 文件中，所有的标签都是相对应的，开头标签为< >，结束标签为</ >，在这两个标签中间添加内容。

有了标签作为文件的主干后，HTML 文件中便可添加属性、数值、嵌套结构等各种类型的内容了。

【实例 3-1】第一个 HTML 网页

读者可以用记事本或任意一个可以进行文本编辑的工具来编写 HTML 代码。下面是一个最基本的 HTML 文档的源代码，程序代码如 ex3_1.html 所示。

ex3_1.html

```
<HTML>
   <HEAD>
      <TITLE>My first page</TITLE>
   </HEAD>
   <BODY>
      我的第一个网页
   </BODY>
</HTML>
```

注意：

用文本编辑工具编写好以后，将这个文件保存成扩展名为 html 类型，再使用浏览器打开这个文件就可以看到结果了。

HTML 文件由标签和被标签标记的内容组成。每个标签都有“<”和“>”围住，以表示这是 HTML 代码而非普通文本，浏览器解析标签后就能产生所需的各种效果。就像一个排版程序，它将网页的内容排成理想的效果。这些标签名称大都为相应的英文单词首字母或缩写，很好记忆。各种标签差别很大，但总的表示形式却大同小异。

HTML 文件以<HTML>开头，以</HTML>结尾，表示这是一个 HTML 格式的网页文件。其中又包括头部(HEAD)和正文(BODY)。

3. HTML 头部及其标签

<HEAD>…</HEAD>，表示这是 HTML 网页的文件头部分，用来说明文件命名和与文件本身的相关信息。通常这部分标签是声明此网页的：默认语言编码、关键字、使用软

件等，个别的标签产生页面动作。在简单的网页中这部分不重要，而较复杂的网页中，比如用 CSS 样式表、JavaScript 语言等，这部分就相当重要。

(1) META 标签

META 标签是 HEAD 部分的一个辅助性标签，它位于 HTML 文档头部的<HEAD>标签和<TITLE>标签之间，它提供用户不可见的信息。

META 标签是一个单标签，可分为两大部分：HTTP 标题信息(http-equiv)和页面描述信息(name)。

① HTTP 标题信息(http-equiv)。这个属性相当于 HTTP 的头文件的作用，用于向浏览器传回一些有用的信息，帮助其正确地显示内容，其基本语法为：

```
<META http-equiv="类型" content="内容">
```

其中类型可以使用以下参数。

- 显示字符集：content-Type

设定页面使用的字符集，即说明主页制作时所使用的文字语言，浏览器会根据这个设置来调用相应的字符集再显示出 page 的内容，例如：

```
<META http-equiv="content-type" content="text/html;charset= GB2132">
```

该 META 标签定义了 HTML 页面所使用的字符集为 GB2132，就是国标汉字码。如果将其中的“charset=GB2312”替换成“BIG5”，则该页面所用的字符集就是繁体中文 Big5 码。当浏览一些国外的站点时，IE 浏览器会提示要正确显示该页面需要下载 xx 语支持。这个功能就是通过读取 HTML 页面 META 标签的 Content-Type 属性而得知需要使用哪种字符集显示该页面的。如果系统里没有安装相应的字符集，则 IE 就提示下载。其他的语言也对应不同的字符集，比如日文的字符集是“iso-2022-jp ”，韩文的是“ks_c_5601”。

对于默认语言，如果不加以说明且网页的内容包含中文，对于默认语言为中文的计算机，IE 会用本机的默认语言打开网页，不会造成网页显示混乱；但如果默认语言不是中文的计算机，则会显示乱码。

- 刷新：refresh

让网页多长时间(秒)刷新自己，或在多长时间后让网页自动链接到其他网页。其用法为：

```
<META http-equiv="refresh" content="30">
```

例如：

```
<META http-equiv="refresh" content="10;url=http://www.njupt.edu.cn">
```

本句将起到网页自动刷新的作用。content=" "内的内容为：刷新延时时间(单位：秒)；打开的网页名称(刷新后打开的网页位置：url=指明的地址和网页文件，可以是绝对地址：http://***/***/*.htm 或相对地址：***/*.htm)。本例为 10 秒，打开 www.njupt.edu.cn 的首页。

- 期限：expires

指定网页在缓存中的过期时间，一旦网页过期，必须到服务器上重新调阅。其用法为：

```
<META http-equiv="expires" content="0">
```

例如：

```
<Meta http-equiv="Expires" Content="Wed, 26 Feb 1997 08:21:57 GMT">
```

必须使用 GMT 的时间格式，或直接设为 0(这个数字表示多少时间后过期)。

- cach 模式：pragma

禁止浏览器从本地机的缓存中调阅页面内容，其用法为：

```
<META http-equiv="pragma" content="No-cach">
```

网页不保存在缓存中，每次访问都刷新页面。这样设定，访问者将无法脱机浏览。

- cookie 设置：set-cookie

浏览器访问某个页面时会将它存在缓存中，下次再次访问时就可从缓存中读取，以提高速度。如果希望访问者每次都刷新广告的图标，或每次都刷新计数器，就要禁用缓存了。通常 HTML 文件没有必要禁用缓存，对于 ASP 等页面，就可以使用禁用缓存，因为每次看到的页面都是在服务器中动态生成的，缓存就失去了意义。如果网页过期，那么存盘的 cookie 将被删除。其用法为：

```
<META http-equiv="set-cookie" content="cookievalue=xxx; expires=Wednesday, 21-Oct-98 16:14:21 GMT; path=/">
```

此处必须使用 GMT 的时间格式。

- 显示窗口的设定：window-target

强制页面在当前窗口以独立页面显示，其用法为：

```
<META http-equiv="widow-target" content="_top">
```

这个属性是用来防止别人在框架里调用自己的页面。其中 content 选项包括_blank、_top、_self、_parent。

- 页面进入与退出：page-enter、page-exit

这个是页面被载入和调出时的一些特效，其用法为：

```
<META http-equiv="page-enter" content="blendTrans(Duration=0.5)">
<META http-equiv="page-exit" content="blendTrans(Duration=0.5)">
```

blendTrans 是动态滤镜的一种，产生渐隐效果。另一种动态滤镜 revealTrans 也可以用于页面进入与退出效果，例如：

```
<META http-equiv="page-enter" content="revealTrans(duration=x, transition=y)">
<META http-equiv="page-exit" content="revealTrans(duration=x, transition=y)">
```

其中，duration 表示滤镜特效的持续时间(单位：秒)； transition 为滤镜类型，表示使用哪种特效，取值为 0~23。

这些用法实际上属于 CSS 滤镜，这部分内容在后面的章节中介绍。

② 页面描述信息(name)。该属性用于说明和描述网页，name 是描述网页的，对应于 content(网页内容)，以便于搜索引擎机器人查找、分类(几乎所有的搜索引擎都会使用网上机器人自动查找 meta 值来给网页分类)。

name 的取值(name="")指定所提供信息的类型，其中有些值是已经定义好的，例如 description(说明)、keywords(关键词)、refresh(刷新)等；也可以指定其他任意值，例如，creationdate(创建日期) 、document ID(文档编号)和 level(等级)等。其基本语法为：

```
<META name="名称" content="内容">。
```

对于有关的用法具体说明如下：

- 关键词：keywords

用于声明网页的关键字，由 Internet 搜索引擎完成关键字索引。正规网站中的主页和关键内容页面应该有关键字，其用法为：

```
<META name="Keywords" content="关键词 1,关键词 2，关键词 3,关键词 4,…">
```

例如：

```
<META name="keywords" content="旅游、计算机知识介绍、网上交友">
```

各关键词间用英文逗号“,”隔开。当数个 META 元素提供文档语言从属信息时，搜索引擎会使用 lang 特性来过滤并通过用户的语言优先参照来显示搜索结果。例如：

```
<META name="Kyewords" lang="EN" content="vacation,greece,sunshine">
<META name="Kyewords" lang="FR" content="vacances,grè:ce,soleil">
```

- 简介：Description

Description 用来描述网站的主要内容，搜索引擎一般也会通过这个属性来检索这个网站，但不显示出来。其用法为：

```
<META name="Description" content="你网页的简述">
```

- 机器人向导：Robots

该属性用来告诉搜索机器人哪些页面需要索引，哪些页面不需要索引。content 的参数有 All、None、Index、Noindex、Follow、Nofollow。默认是 All。其用法为：

```
<META name="Robots" content="All|None|Index|Noindex|Follow|Nofollow">
```

许多搜索引擎都通过 robot/spider 来搜索网站，这些 robot/spider 就要用到 META 元素的一些特性来决定怎样登录。其中可用选项的含义如下。

- all：文件将被检索，且页面上的链接可以被查询；
- none：文件将不被检索，且页面上的链接不可以被查询(和 "noindex, no follow" 起相同作用)；
- index：文件将被检索(让 robot/spider 登录)；
- follow：页面上的链接可以被查询；

➢ noindex：文件将不被检索，但页面上的链接可以被查询(不让 robot/spider 登录)；
➢ nofollow：文件将不被检索，页面上的链接可以被查询(不让 robot/spider 顺着此页的连接往下查找)。

● 作者：Author

标注网页的作者或制作组，其用法为：

```
<META name="Author" content="张三，abc@sina.com">
```

注意：content 也可以是：网页制作者或网页制作者的制作组的名字，或 E-mail

● 版权：Copyright

该属性用于标注版权，其用法为：

```
<META name="Copyright" content="本页版权归 njupt 所有。All Rights Reserved">
```

● 编辑器：Generator

编辑器的说明，其用法为：

```
<META name="Generator" content="PCDATA|FrontPage|">
```

注意：content="你所用编辑器" ，其中双引号内可以填写 frontpage 等。

● 重访：revisit-after

设定重新访问的时间间隔，以天为单位，其用法为：

```
<META name="revisit-after" content="7 days" >
```

以上设置的以 7 天为间隔再次访问本网页。

③ 其他标签：

● 方案：scheme

当 name 用于解释 content 内容的含义时，可以使用“scheme”这个标签，其用法如下：

```
<META scheme="ISBN" name="identifier" content="0-14-043205-1" />
```

这里声明了一个 ISBN 的标识符，它的内容为 0-14-043205-1。

● 链接：LINK

链接到某个文件，其用法为：

```
<LINK href="soim.ico" rel="Shortcut Icon">
```

如果用户把某个网站保存在收藏夹中后，也许会发现它带着一个小图标，如果再次单击进入之后还会发现地址栏中也有个小图标。如果在<head>中加上这段话，就能轻松实现这一功能。<LINK> 用来将目前文件与其他 URL 进行链接，但不会有链接按钮用于<head>标签间，例如：

```
<LINK href="URL" rel="relationship">
<LINK href="URL" rev="relationship">
```

- 基链接：BASE

加入网页基链接属性，其用法为：

```
<BASE href="http://www.njupt.edu.cn/" target="_blank">
```

加了上面的语句后，网页上所有的相对路径在链接时都会自动在前面加上“http://www.njupt.edu.cn/”。其中 target="_blank"是链接文件在新的窗口中打开，也可以做其他设置。将“_blank”改为“_parent”是链接文件将在当前窗口的父级窗口中打开；改为“_self”链接文件在当前窗口(帧)中打开；改为“_top”链接文件全屏显示。

(2) 网页标题(TITLE)

网页的标题概括了网页的内容，能让用户迅速了解网页的大意。在文件头部定义的标题内容不在浏览器窗口中显示，而是在浏览器的标题栏中显示。尽管头部定义的信息很多，但能在浏览器标题栏中显示的信息只有标题。用法如下：

```
<TITLE>XXX 网页的标题</TITLE>
```

(3) 文档主体部分(BODY)

<BODY>…</BODY>是网页的正文部分。HTML 文件主体标签的格式为：

```
<BODY bgcolor="颜色值" background="文件名" text="颜色值" link="颜色值" vlink="颜色值" 
alink="颜色值">…</body>
```

其中可以设置的属性如表 3-3 所示。

表 3-3　文档主体部分可以设置的属性值

属　　性	描　　述
link	设定页面默认的超链接颜色
alink	设定鼠标正在单击时的超链接颜色
vlink	设定访问后超链接文字的颜色
background	设定页面背景图像
bgcolor	设定页面背景颜色
leftmargin	设定页面的左边距
topmargin	设定页面的上边距
bgproperties	设定页面背景图像为固定，不随页面的滚动而滚动
text	设定页面文字的颜色

在输入颜色值时，也可以是以 0~9 及 a~f 来表示的，例如：#00FF13。颜色是由“red”、“green”和“blue”三原色组合而成的，在 HTML 中对颜色的定义是用十六进位的，对于三原色 HTML 分别给予两个十六进位去定义，也就是每个原色可有 256 种彩度，故此三原色可混合成 16777216 种颜色。

颜色的标识顺序为 RGB，即红绿蓝，每两位代表一种颜色，颜色值的输入可以参考“颜色值表”。

【实例 3-2】一个简单的 HTML 网页

写一个具备上述基本结构的网页，要求网页标题和网页正文的文本内容自定，网页背景为黄色(yellow)，文本颜色为灰色(gray)，超链接被访问前后和被访问的瞬间颜色分别为蓝(blue)、红(red)、绿(green)，程序代码如 ex3_2.html 所示。

ex3_2.html

```
<HTML>
  <HEAD>
    <TITLE>My first page</TITLE>
  </HEAD>
  <BODY bgcolor="yellow" text="gray" link="blue" vlink="red" alink="green">
    <a href="连接的对象">连接的文字</a>
  </BODY>
</HTML>
```

要达到本例的要求，需要对 BODY 的属性进行正确的设置，还应注意正确写出网页的完整结构。

注意：

HTML 代码在书写时不区分大小写，也不要求在书写时缩进，但为了程序的易读性，建议网页设计者使标签的首尾对齐，内部的内容向右缩进几格。

3.3　HTML 的基本语法

3.3.1　标题和段落

1. 标题文字标签

一般文章都有标题、副标题、章和节等结构，HTML 中也提供了相应的标题标签<Hn>，其中 n 为标题的等级别。HTML 总共提供六个等级的标题，n 越小，标题字号就越大。标题文字的格式为：

```
<Hn align=对齐方式>标题文字</Hn>
```

属性 align 用来设置标题在页面中的对齐方式，可以设置为 left(左对齐)，right(右对齐)，center(居中对齐)。

属性 n 用来指定标题文字的大小。n 可以取 1～6 的整数值，取 1 时文字最大，取 6 时文字最小。

与用<TITLE>…</TITLE>所定义的网页标题不同，标题格式显示在浏览器窗口中，而不显示在浏览器的标题栏中。

【实例 3-3】标题文字的用法

程序代码如 ex3_3.html 所示。

ex3_3.html

```
<HTML>
  <HEAD>
    <TITLE>设置标题</TITLE>
  <HEAD>
  <BODY>
    <H1>第 1 级标题(H1)</H1>
    <H2>第 2 级标题(H2)</H2>
    <H3>第 3 级标题(H3)</H3>
    <H4 align=left>第 4 级标题(H4)(居左)</H4>
    <H5 align=right>第 5 级标题(H5)(居右)</H5>
    <H6 align=center>第 6 级标题(H6)(居中)</H6>
  </BODY>
</HTML>
```

该例分 6 行写出了六级标题，在后三行加上了对齐属性并分别赋予 3 个不同的属性值。在浏览器中运行后可得结果如图 3-13 所示。

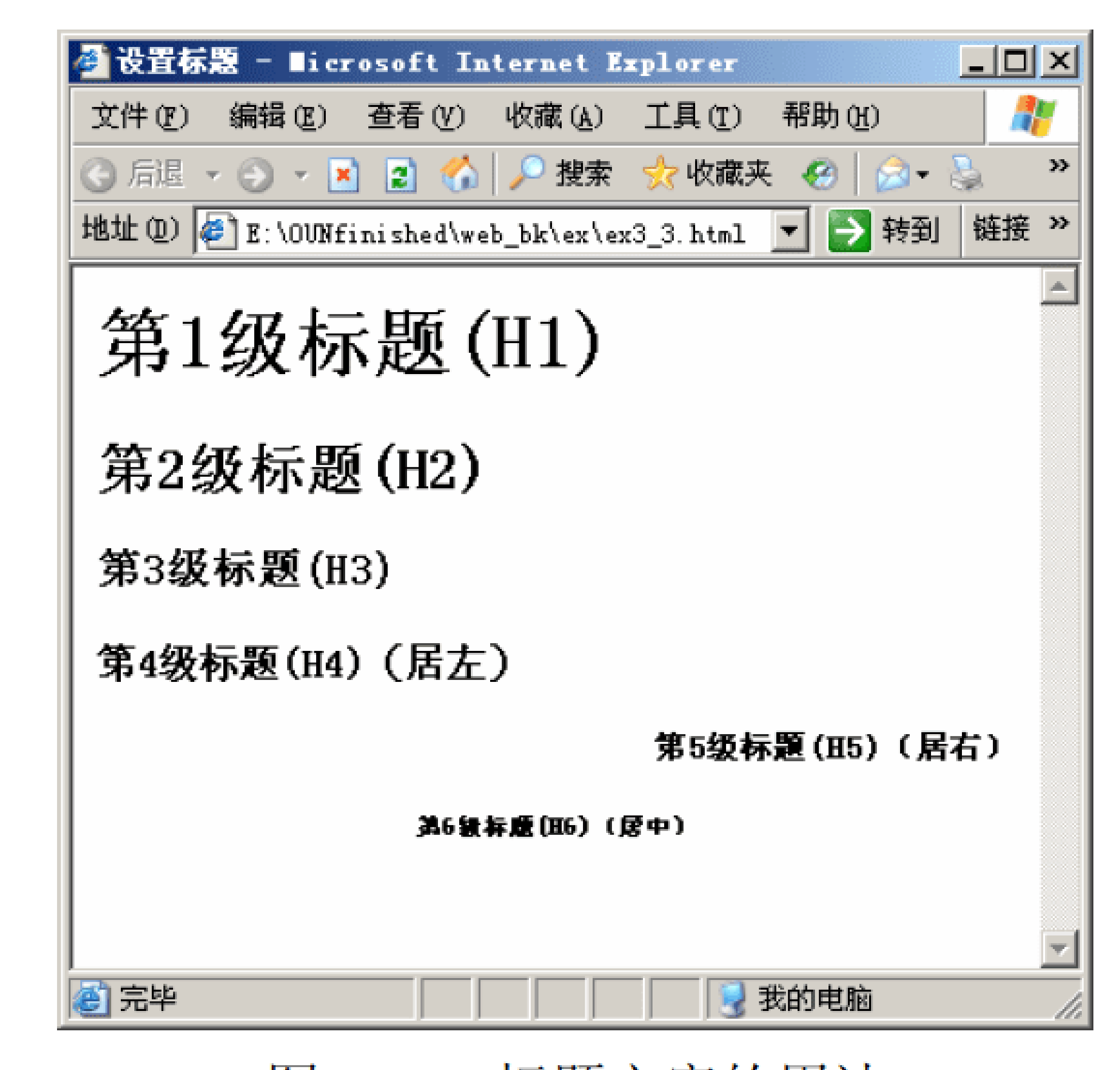


图 3-13　标题文字的用法

2. 段落标签

常用的段落标签包括
、<P>、<PRE>、<DIV>等，下面逐一介绍。

(1) 强行换行标签

在编写 HTML 文件时，我们不必考虑太细的设置，也不必理会段落过长的部分会被浏览器切掉。因为，在 HTML 语言规范里，每当浏览器窗口被缩小时，浏览器会自动将右边的文字转折至下一行。所以，编写者对于明确需要断行的地方，应加上
标签。
放在一行的末尾，可以使后面的文字、图片、表格等显示于下一行，而又不会在行与行之间留下空行，即强制文本换行。这使
成为最常用的标签之一。强制换行标签的格式为：

```
文字<BR>
```

以下的例子说明了这个标签的用法。

【实例 3-4】强行换行标签的用法

程序代码如 ex3_4.html 所示。

ex3_4.html

```
<HTML>
  <HEAD>
```

```
        <TITLE>换行示例</TITLE>
    </HEAD>
    <BODY>
        登鹳雀楼<BR>白日依山尽，<BR>黄河入海流。<BR>欲穷千里目，<BR>更上一层楼
    </BODY>
</HTML>
```

运行后可以看到如图 3-14 所示的结果，虽然在源文件中书写的时候是放在一行之中的文本，因为使用了强行换行标签
，而使得实际显示的时候出现了换行的效果。



图 3-14　强行换行标签的用法

(2) 设置段落标签<P>

为了排列得整齐、清晰，文字段落之间，我们常用<P></P>来做标签。文件段落的开始由<P>来标示，段落的结束由</P>来标示。<P>标签不但能使后面的文字换到下一行，还可以使两段之间多一空行。由于一段的结束意味着新一段的开始，所以使用<P>也可省略结束标签。

一个强制换段标签<P>可以看作是两个强制换行标签

。

设置段落标签的格式为：

```
<P align=对齐方式>文字</P>
```

其中属性 align 用来设置段落的对齐方式，可以为 left、center 或 right，分别表示居左、居中或居右。段落设置默认为 left。

【实例 3-5】段落标签的用法

程序代码如 ex3_5.html 所示。

ex3_5.html

```
<HTML>
    <HEAD>
        <TITLE>段落标签示例</TITLE>
    </HEAD>
    <BODY>
    <P ALIGN=CENTER>
        登黄鹤楼<BR>白日依山尽，<BR>黄河入海流。<BR>欲穷千里目，<BR>更上一层楼。
    </P>
    </BODY>
</HTML>
```

运行后可以看到如图 3-15 所示的结果，图中所有文字均居中显示了。



图 3-15　强行换行标签的用法

(3) 显示预排格式标签<PRE>

当用其他编辑工具编排好了一段文字后，其中很可能有一些 HTML 文件不支持控制符号，如 Enter 键、多个空格、<Tab>键等。如果希望在浏览网页时仍保留在编辑工具中已经排好的段落格式，可以使用<PRE>标签将预先排好的格式保留下来。显示预排格式标签的格式为：

```
<PRE>  预先排好的格式  </PRE>
```

在预排格式中，仍可以用 HTML 语言对文字的格式进行设置，如文字的颜色、大小等。

【实例 3-6】显示预排格式标签的用法

程序代码如 ex3_6.html 所示。

ex3_6.html

```
<HTML>
  <HEAD>
    <TITLE>  显示预排格式  </TITLE>
  </HEAD>
  <BODY>
    <PRE>
      <FONT size=3 color=blue>
        <H1>
          <FONT color=purple>
            <P align=left>           唐诗二首</P></FONT></H1><FONT face=黑体 color=black>
         赋得古原草送别              长      相      思</FONT>
           唐·白居易                  唐·白居易
     离离原上草，一岁一枯荣。          汴水流，泗水流，
     野火烧不尽，春风吹又生。          流到瓜洲古渡头。
     远芳侵古道，晴翠接荒城。          吴山点点愁。
     又送王孙去，萋萋满别情。          思悠悠，恨悠悠，
                                      恨到归时方始休。
                                      月明人依楼。
      </FONT>
    </PRE>
  </BODY>
</HTML>
```

运行后可以看到如图 3-16 所示的结果，图中所有文字均按照原来所书写的格式显示出来了。

(4) 分区显示标签<DIV>

文本块、一段文字或标题在网页上的布局都有左对齐、居中和右对齐 3 种方式。在标签中使用 align 属性，其属性取值分别为 left、center、right。可以设置布局的标签有：<P>…</P>、<HN>…</HN>、<DIV>…</DIV>、<CENTER>…</CENTER>等。当在许多段落中设置对齐方式时，常使用<DIV>…</DIV>标签。分区显示标签的格式为：

```
<DIV align=left|center|right>文本或图像</DIV>
```

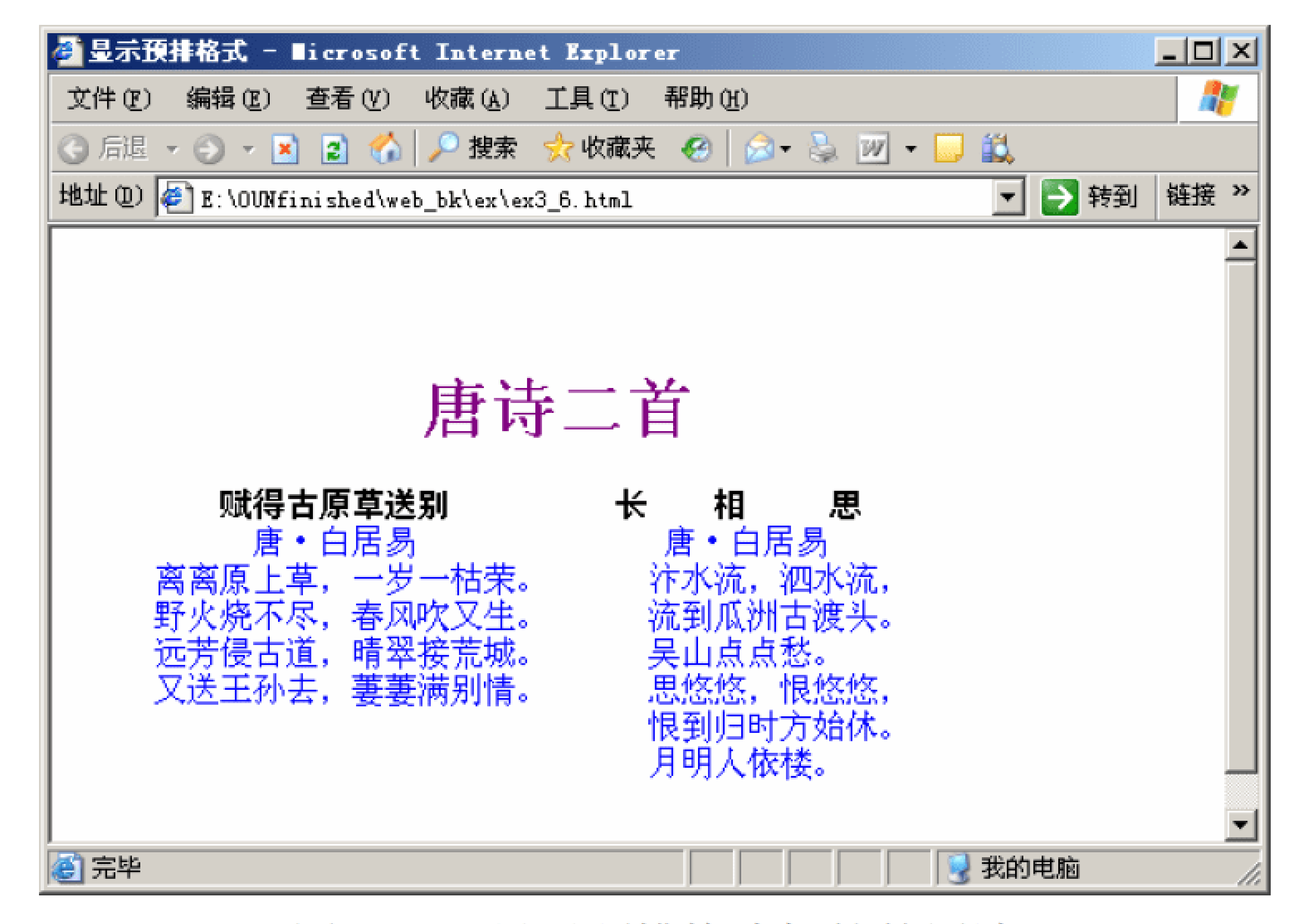


图 3-16　显示预排格式标签的用法

【实例 3-7】分区显示标签<DIV>的用法

程序代码如 ex3_7.html 所示。

ex3_7.html

```
<HTML>
  <BODY>
    <CENTER>
      <H2>分区显示标签的应用</H2>
    </CENTER>
    <DIV align=center>居中 center<BR>居中<BR>center</DIV>
    <DIV align=left>居左 left<BR>居左<BR>left</DIV>
    <DIV align=right>居右 right<BR>居右<BR>right</DIV>
  </BODY>
</HTML>
```

运行后可以看到如图 3-17 所示的结果，图中 DIV 标签及 align 属性控制了显示的位置。

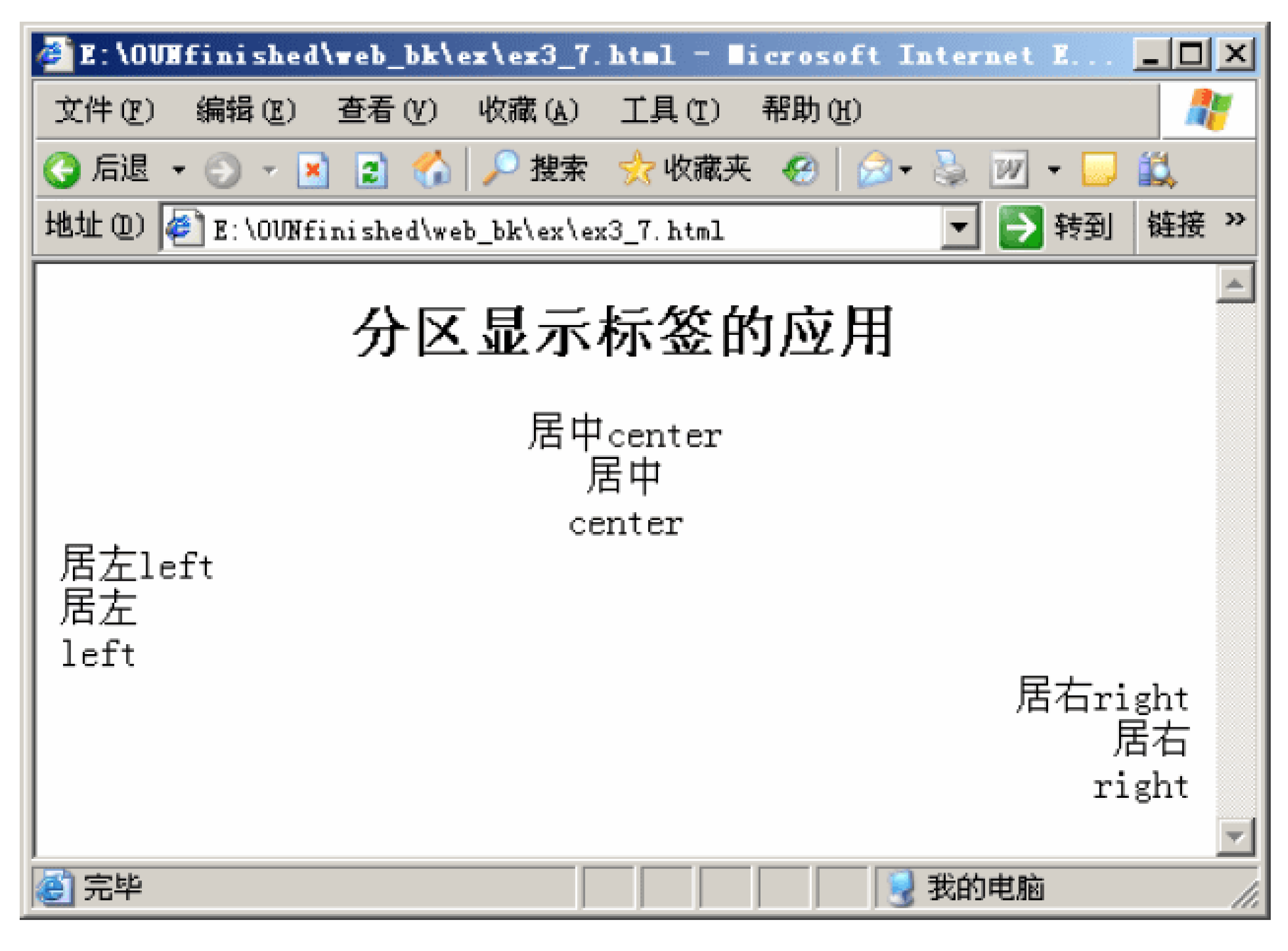


图 3-17　分区显示标签<DIV>的用法

(5) 水平线

在页面中插入一条水平标尺线(Horizontal Rules)，可以使不同功能的文字分隔开，看起来整齐、明了。当浏览器解释到 HTML 文件中的<HR>标签时，会在此处换行，并加入

一条水平线段。线段的样式由标识的参数决定。水平线标签的格式为：

```
<HR align=对齐方式 size=横线粗细 width=横线长度 color=横线颜色 noshade>
```

其中，各属性的作用如下。

- align：设定横线放置的位置，可选择 left(居左)、right(居右)或 center(居中)。
- size：设定线条粗细，以像素为单位，默认为 2。
- width：设定线段长度，可以是绝对值(以像素为单位)或相对值(相对于当前窗口的百分比)。

注意：

所谓的绝对值，是指线段的长度是固定的，不随窗口尺寸的改变而改变。所谓相对值，是指长度相对于窗口的宽度而定，窗口的宽度改变时，线段的长度也随之增减，默认值为 100%，即始终填满当前窗口。

color 属性设定线条颜色，默认为黑色。可以采用颜色的名称。颜色可以用相应英文单词或以“#”引导的一个十六进制数代码来表示。

noshade 属性设定线条为平面显示(没有三维效果)，若默认则有阴影或立体效果。

【实例 3-8】水平线的用法

程序代码如 ex3_8.html 所示。

ex3_8.html

```
<HTML>
  <HEAD> <TITLE> 水平线段标签的应用 </TITLE> </HEAD>
  <BODY>
     <CENTER><H3>水平线</H3></CENTER>
     <HR>
     <HR align=left size=6 width=320>
     <HR align=center size=8 width=60% color=blue>
     <HR align=right size=8 width=360 color=red>
     <HR size=4 width=80% color=#CD061F>
     <HR size=5 noshade>
     <HR width=70% noshade>
  </BODY>
</HTML>
```

运行后的显示如图 3-18 所示，从图中可以看到不同颜色、宽度和样式的水平线。在浏览时，如果改变一下窗口的大小，可以看到线段的变化效果。

3.3.2　文字标签

1. 设置文字的大小

在网页中为了增强页面的层次，其中的文字可以用不同的大小、字体、字型、颜色。用<FONT>标签设置字号。设置文字大小的格式为：

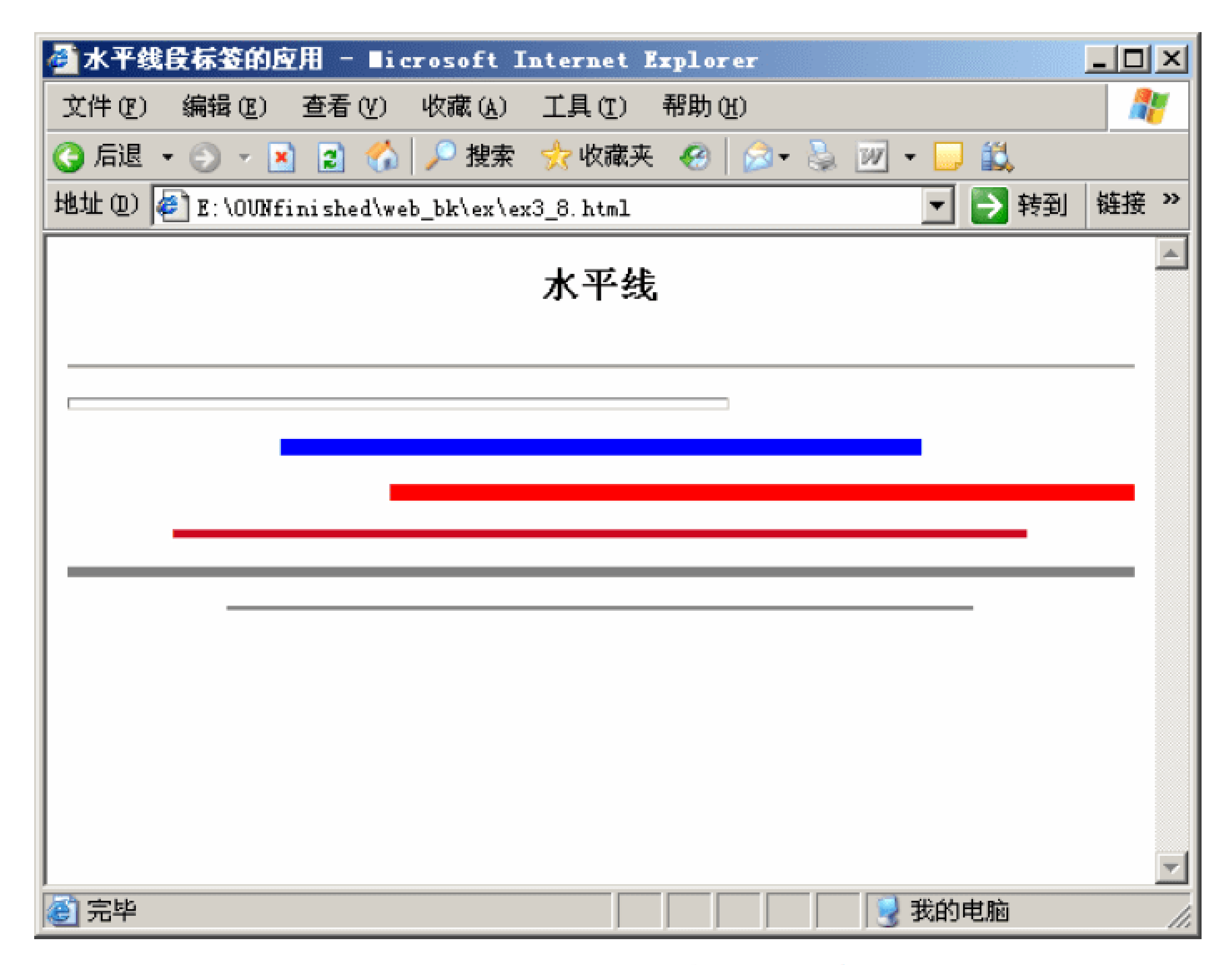


图 3-18　水平线的用法

```
<FONT size=数字  face=字体名  color=颜色>  被设置的文字  </FONT>
```

<FONT>标签可设定文字的字体、字号和颜色。其中各属性的作用如下。

- size：用来设置文字的大小。数字的取值范围为 1~7，size 取 1 时最小，取 7 时最大。
- face：用来设置字体。如黑体、宋体、楷体_GB2312、隶书、Times New Roman 等。
- color：用来设置文字颜色。

【实例 3-9】用<FONT>设置文字的大小

程序代码如 ex3_9.html 所示。

ex3_9.html

```
<HTML>
  <BODY>
    <CENTER>
      <FONT size="1">一号字</FONT><BR>
      <FONT size="2">二号字</FONT><BR>
      <FONT size ="3">三号字</FONT><BR>
      <FONT size ="4">四号字</FONT><BR>
      <FONT size ="5">五号字</FONT><BR>
      <FONT size ="6">六号字</FONT><BR>
      <FONT size ="7">七号字</FONT><BR>
    </CENTER>
  </BODY>
</HTML>
```

运行后的显示如图 3-19 所示，从图中可以看到使用<FONT>的不同 size 属性所得到的字体。

<FONT>和<Hn>标签都可以设置文字的大小，但是两者是不同的，二者的区别如表 3-4 所示。

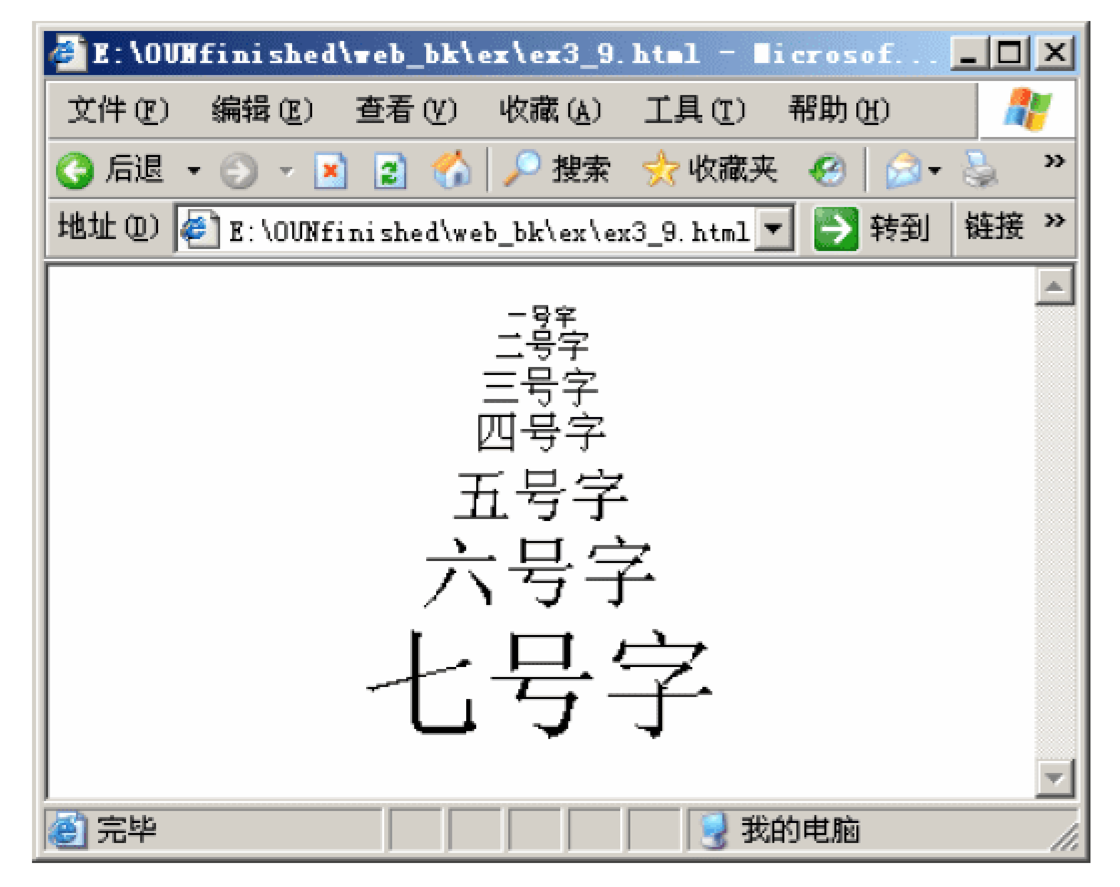


图 3-19　用<FONT>设置文字的大小

表 3-4　<FONT>与<Hn>标签的区别

标　签	语　法	施 加 对 象	文字大小 n 的取值	字 体 加 粗
<FONT>	<FONT size=n>文字</FONT>	一段文章、语句、短语	n=1～7，取 1 时最小，7 时最大	不自动加粗
<Hn>	<Hn> 文字 </Hn>	标题	n=1～6，取 6 时最小，1 时最大	自动加粗

2. 设置文字的字体

当文字为汉字时，格式中的“字体名”可以为宋体、幼圆、隶书、楷体_GB2312、黑体、仿宋_GB2312 等。当文字为英文时，字体名可以为 Times New Roman 等约 50 种字体。其实，这里的字体名与 Word 中的“字体”工具栏中的字体名相同。

【实例 3-10】用<FONT>设置文字的字体

程序代码如 ex3_10.html 所示。

ex3_10.html

```
<HTML>
  <HEAD>
     <TITLE>设置字体</TITLE>
  </HEAD>
  <BODY>
    <FONT size=4>
      <FONT face=幼圆> 幼圆 </FONT><BR>
      <FONT face=隶书> 隶书 </FONT><BR>
      <FONT face=楷体> 楷体 </FONT><BR>
      <FONT face=黑体> 黑体 </FONT><BR>
      <FONT face=仿宋> 仿宋 </FONT><BR>
      <FONT face=宋体> 宋体 </FONT><BR>
      <FONT face=方正舒体> 方正舒体 </FONT><BR>
      <FONT face=华文彩云> 华文彩云 </FONT> <BR>
      <FONT face=华文琥珀> 华文琥珀 </FONT> <BR>
      <FONT face=Times New Roman> Times New Roman </FONT> <BR>
    </FONT>
```

```
  </BODY>
</HTML>
```

运行后的浏览器显示如图 3-20 所示，从图中可以看到使用<FONT>的不同 face 属性所得到的字体。



图 3-20　用<FONT>设置文字的字体

3. 设置文字的字型

字型就是文字的风格，如加粗、斜体、带下划线、上标、下标等。对字型的控制标签如表 3-5 所示。

表 3-5　设置各种字型的标签

标 签 格 式	字 体 结 果
<B>受影响的文字</B>	加粗
<I>受影响的文字</I>	斜体
<U>受影响的文字</U>	带下划线
<TT>受影响的文字</TT>	标准打印机字体
<STRIKE>受影响的文字</STRIKE>	带删除线
_{受影响的文字}	下标
^{受影响的文字}	上标
<BIG>受影响的文字</BIG>	大字体文本
<SMALL>受影响的文字</SMALL>	小字体文本

注意：

其中的<U>、<TT>、< STRIKE >、<BIG>这四个标签在 HTML 5 中不支持。此外，有些虽然名称相同，但在 HTML 5 中进行了重新定义。详细的用法最好参考有关手册或使用浏览器进行测试。

【实例 3-11】设置文字的字型

程序代码如 ex3_11.html 所示。

ex3_11.html

```
<HTML>
  <HEAD>
    <TITLE> 设置字型 </TITLE>
  </HEAD>
  <BODY>
    正常字体   <B>加粗体</B> <BR>
    正常字体   <I>斜体</I><BR>
    正常字体   <U>下划线</U><BR>
    正常字体   <TT>标准打印机字体</TT><BR>
    正常字体   <STRIKE>带删除线</STRIKE><BR>
    正常字体   <SUB>下标</SUB> <SUP>上标</SUP><BR>
    正常字体   <BIG>大字体文本</BIG><BR>
```

```
        正常字体  <SMALL>小字体文本</SMALL><BR>
    </BODY>
</HTML>
```

运行后的浏览器显示如图 3-21 所示，从图中可以看到使用不同字型所得到的结果。

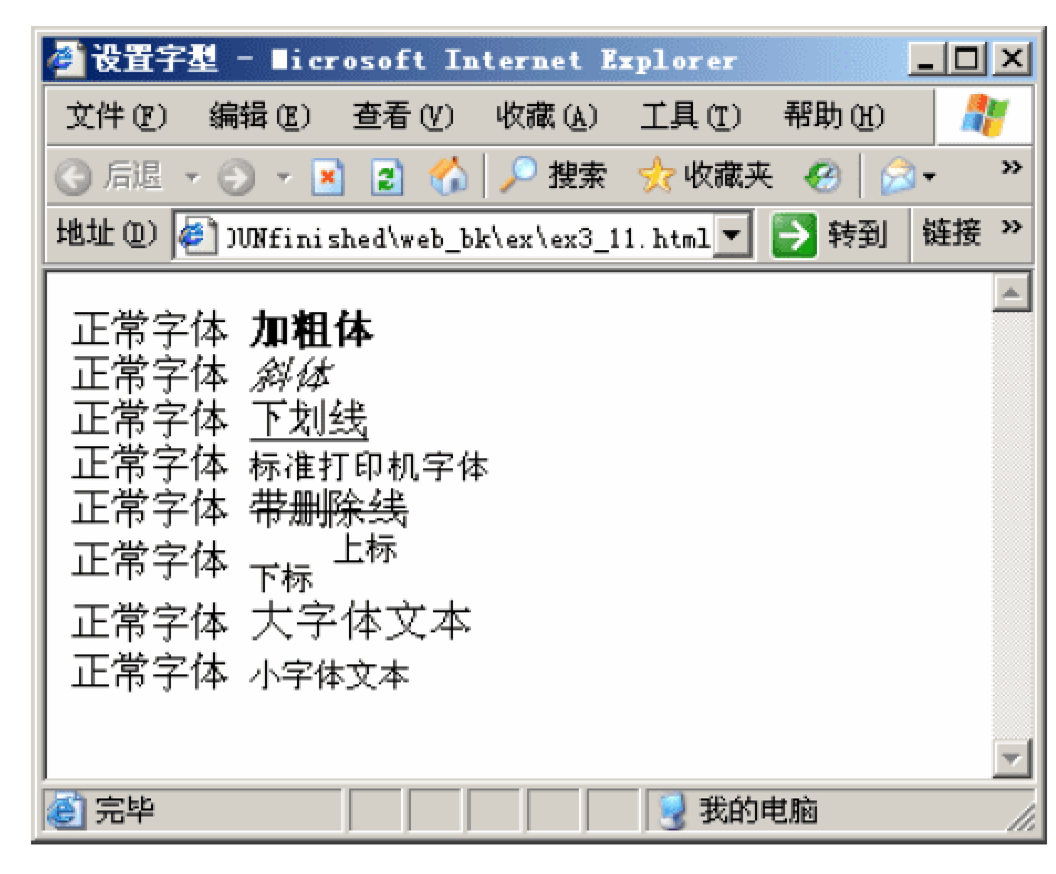


图 3-21　设置文字的字型

4. 设置文字的颜色

设置文字的颜色有两种方法。

(1) <BODY>标签中的 text 属性

利用<BODY>标签中的 text 属性，可以设定整个网页文字的颜色。格式如下：

```
<BODY text=颜色>受影响的文字</BODY>
```

<BODY>与</BODY>标签所包含的内容是 HTML 文件的主体，所以它带的参数对整个文件起作用。

【实例 3-12】利用<BODY>设置文字的颜色

程序代码如 ex3_12.html 所示。

ex3_12.html

```
<html>
    <head>
        <title> 用 body 标签设定文字的颜色 </title>
    </head>
    <body text=blue>
        <h3>春夜喜雨</h3>
        <font face=宋体  size=4>杜   甫(唐)</font><br><br>
        <font face=楷体>
            好雨知时节，当春乃发生。<br>
            随风潜入夜，润物细无声。<br>
            野径云俱黑，江船火独明。<br>
            晓看红湿处，花重锦官城。<br>
        </font>
    </body>
</html>
```

运行后的浏览器显示如图 3-22 所示，从图中可以看到使用<BODY>中的 text=blue 属性设置可以得到的蓝色文字。

(2) <FONT>标签中的 color 属性

利用<FONT>标签中的 color 属性，可以设定网页、段落、短语、词或字的颜色。其格式为：

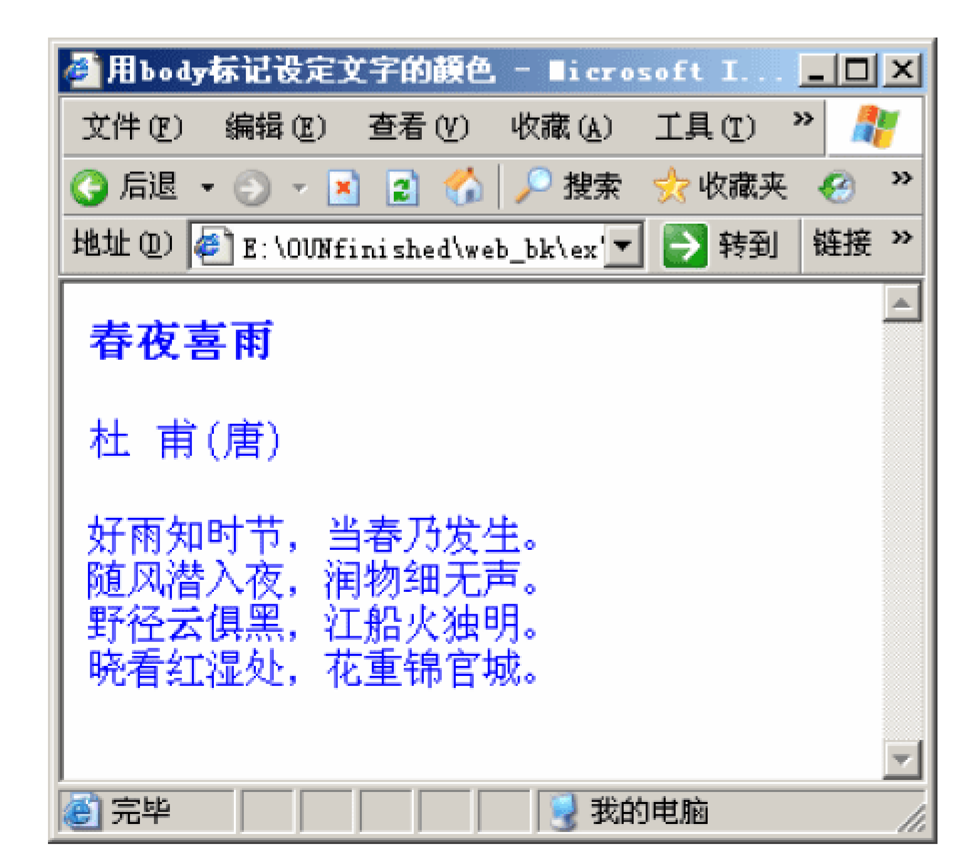


图 3-22　利用<BODY>设置文字的字型

```
<FONT color=颜色>受影响的文字</FONT>
```

<FONT>标签直接作用于该标签与紧跟其后的</FONT>之间的文字，可在一个 HTML 文件中的多处进行设置。

【实例 3-13】利用<FONT>设置文字的颜色

程序代码如 ex3_13.html 所示。

ex3_13.html

```
<HTML>
  <HEAD>
    <TITLE>用 FONT 标签设定文字的颜色
    </TITLE>
  </HEAD>
  <BODY>
    <FONT color=white>白色</FONT> <FONT color=yellow>黄色</FONT><BR>
    <FONT color=crimson>深红色</FONT> <FONT color=greenyellow>黄绿色</FONT><BR>
    <FONT color=brown>棕色</FONT> <FONT color=cyan>青色</FONT><BR>
    <FONT color=gray>灰色</FONT> <FONT color=green>绿色</FONT><BR>
    <FONT color=ivory>乳白色</FONT> <FONT color=orange>橘黄色</FONT><BR>
    <FONT color=black>黑色</FONT> <FONT color=blue>蓝色</FONT><BR>
    <FONT color=pink>粉红色</FONT> <FONT color=red>红色</FONT><BR>
    <FONT color=dodgerblue>水蓝色</FONT> <FONT color=lavender>淡紫色</FONT>
  </BODY>
</HTML>
```

运行后的浏览器显示如图 3-23 所示，读者可以自行运行光盘中的实例代码后看到使用<FONT>来设置不同的文字颜色的效果。

当<BODY>与<FONT>标签同时对文字颜色进行定义且两者定义的颜色不同时，此时采用就近原则，即<FONT>标签所规定的颜色优先。



图 3-23　利用<FONT>设置文字的颜色

3.3.3　列表

HTML 提供了可以生成列表的元素，列表分为无序列表、有序列表和定义列表。带序号标志(如数字、字母等)的表项就组成有序列表，否则为无序列表；定义列表由用户自己进行定义。列表可以嵌套，而且不同类型的列表可以一起使用。

1. 无序列表

无序列表中每一个表项的前面是项目符号(如●、■等)。建立无序列表使用<UL>标签和<LI>表项标签，其中 UL 为 Unordered List 的缩写，而 LI 为 List Item 的缩写。其使用格

式为：

```
<UL type="符号类型">
    <LI type="符号类型 1"> 第一个列表项
     <LI type="符号类型 2"> 第二个列表项
      …
</UL>
```

值得注意的是，<LI>标签是单标签。即一个表项的开始，就是前一个表项的结束。

从浏览器最终显示的效果看，无序列表的特点是，列表项目作为一个整体，与上下段文本间各有一行空白；表项向右缩进并左对齐，每行前面有项目符号。

type 指定每个表项左端的符号类型，可取值为 disc(实心圆点)、circle(空心圆点)、square(方块)，也可自己设置图片，方法有如下两种。

(1) 在<UL>后指定符号的样式

在<UL>后指定符号的样式，可设定直到</UL>的加重符号。例如：

```
<UL   type="disc">                    符号为实心圆点●
<UL   type="circle">                  符号为空心圆点○
<UL   type="square">                  符号为方块■
<UL   img src="YourGraph.gif">        符号为指定的图片文件
```

(2) 在<LI>后指定符号的样式

在<LI>后指定符号的样式，可以设置从该<LI>起直到</UL>的项目符号。其格式就是将前面的 UL 换为 LI。

【实例 3-14】无序列表的应用

程序代码如 ex3_14.html 所示。

ex3_14.html

```
<HTML>
  <HEAD>
    <TITLE>无序列表</TITLE>
  </HEAD>
  <BODY>
    <P align=center><FONT color=blue size=4><B>无序列表的使用</B></FONT></P>
    <UL>
      <LI type="circle">圆圈
      <LI type="square">方块
      <LI type="disc">圆点
      <LI>默认样式
    </UL>
    <UL type="circle">
      <LI>上层定义 1
      <LI>上层定义 2
      <LI>上层定义 3
      <LI>上层定义 4
      <LI type="square">本地定义方块
```

```
        <LI>上层定义 5
      </UL>
    </BODY>
</HTML>
```

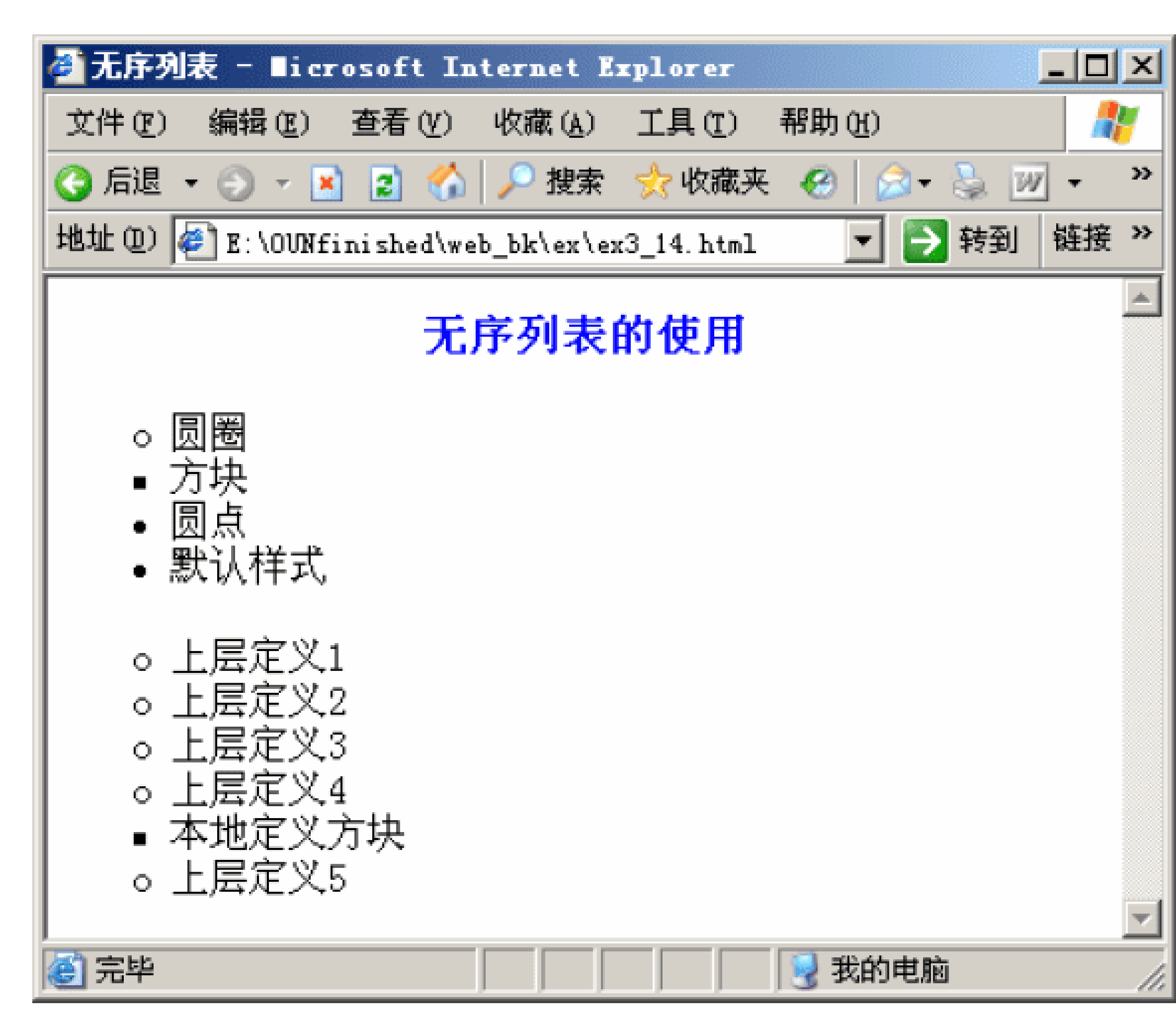


图 3-24　无序列表的应用

运行后的浏览器显示如图 3-24 所示，从图中可以看到无序列表的三种定义方式以及在统一定义后的效果。

2. 有序列表

通过带序号的列表可以更清楚地表达信息的顺序。使用<OL>标签可以建立有序列表，表项的标签仍为<LI>；其中 OL 为 Ordered List 的缩写，LI 为 List Item 的缩写。其格式为：

```
<OL type="符号类型">
<LI type="符号类型 1"> 表项 1
<LI type="符号类型 2"> 表项 2
…
</OL>
```

在浏览器中显示时，有序列表整个表项与上下段文本之间各有一行空白；列表项目向右缩进并左对齐；各表项前带顺序号。

可以改变有序列表中的序号种类，利用<OL>或<LI>中的 type 属性可设定 5 种序号：数字、大写英文字母、小写英文字母、大写罗马字母和小写罗马字母。默认的序号标签是数字。

在<OL>后指定符号的样式，可设定直到</OL>的表项加重记号。格式为：

```
<OL   type="1">      序号为数字
<OL   type="A">      序号为大写英文字母
<OL   type="a">      序号为小写英文字母
<OL   type="I">      序号为大写罗马字母
<OL   type="i">      序号为小写罗马字母
```

在<LI>后指定符号的样式，可设定该表项前的加重记号。格式只需把上面的 OL 改为 LI。

【实例 3-15】有序列表的应用

程序代码如 ex3_15.html 所示。

ex3_15.html

```
<HTML>
  <HEAD>
    <TITLE>有序列表</TITLE>
  </HEAD>
```

```
    <BODY>
      <CENTER>
        <OL>
          <LI><FONT SIZE="7">第一列</FONT>
          <LI><FONT SIZE="7">第二列</FONT>
          <LI><FONT SIZE="7">第三列</FONT>
        </OL>
      </CENTER>
    </BODY>
  </HTML>
```

图 3-25　有序列表的应用

运行后的浏览器显示如图 3-25 所示，从图中可以看到简单有序列表的显示效果。

【实例 3-16】不同种类有序列表的应用

程序代码如 ex3_16.html 所示。

ex3_16.html

```
  <HTML>
    <HEAD><TITLE>设置列表的种类</TITLE></HEAD>
    <BODY>
      <P align=center><FONT color=blue size=6>设置列表的种类</FONT></P>
      <OL type="a">
        <LI>a 类 1
        <LI>a 类 2
      </OL>
      <OL type="I">              <!--   注意"I"为大写   -->
        <LI>I 类 1
        <LI type="i">i 类 2     <!--   注意"i"为小写   -->
      </OL>
    </BODY>
  </HTML>
```

运行后的浏览器显示如图 3-26 所示，从图中可以看到不同种类的有序列表的使用效果。

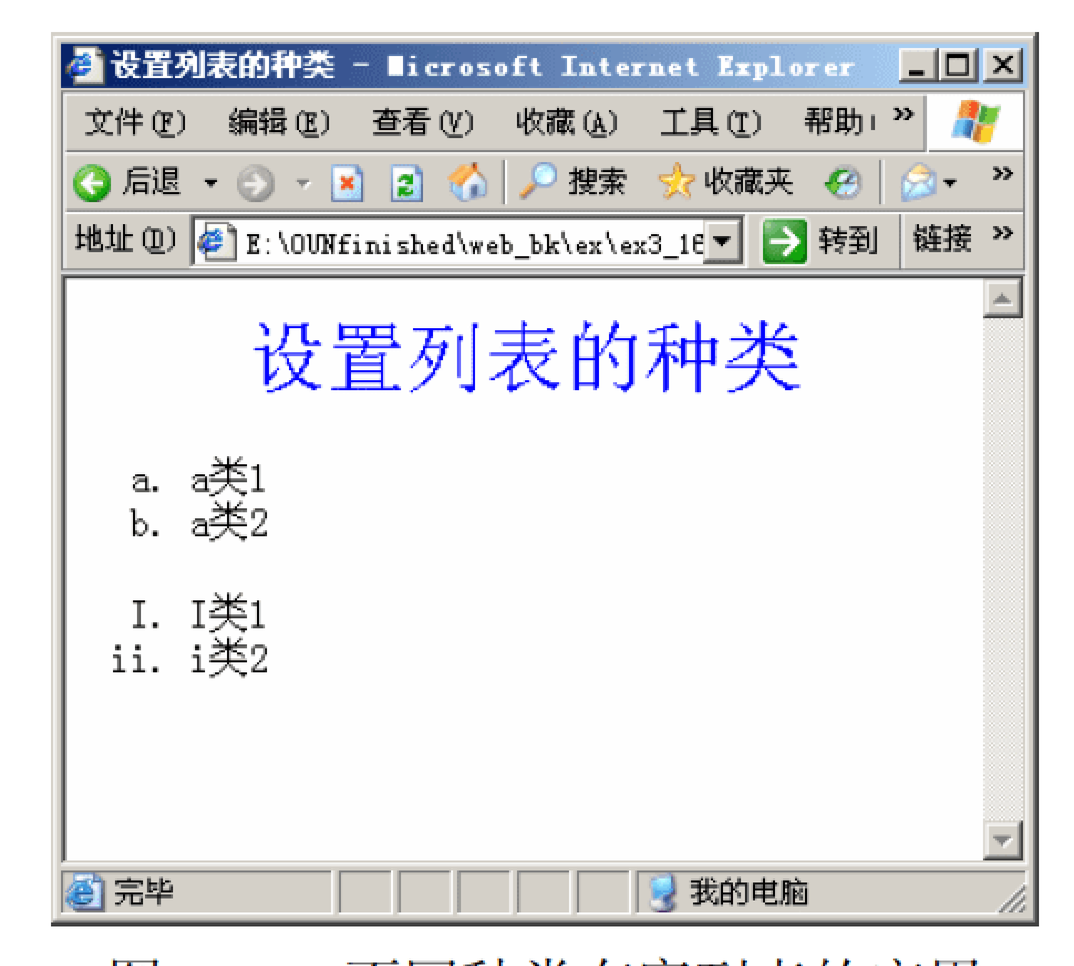


图 3-26　不同种类有序列表的应用

3. 定义列表

用项目列表表示单词或语句，使之具有交互凹进的特点。定义项目列表使用标签<DL>、<DT> 和 <DD>；其中定义列表(DL)为 Definitions List 的缩写，定义术语(DT)为 Definition Term 的缩写，定义描述(DD)为 Definition Description 的缩写。

定义术语(DT)中只能包含 inline 类型的元素，而定义描述(DD)中可以包括任何类型的 HTML 元素，甚至是列表元素。

定义列表中包含了若干组定义，每组由其相关的“术语”及其“描述”构成；浏览器

在处理时，通常会给出特定的缩进格式。<DT>往往用于定义单词，<DD>用于定义语句。由<DT>定义的项目会自动换行左对齐，但项目之间没有空行。格式为：

```
<DL>
<DT> 定义单词 1
<DD> 单词 1 的说明
<DT> 定义单词 2
<DD> 单词 2 的说明
…
</DL>
```

与<LI>标签一样，<DT>、<DD>也为单标签。

【实例 3-17】定义列表的应用

程序代码如 ex3_17.html 所示。

ex3_17.html

```
<HTML>
  <HEAD><TITLE>定义列表</TITLE></HEAD>
  <BODY>
    <P align=center><FONT color=blue size=4><B>***问题</B></FONT></P>
    以上问题可分为 4 个方面：
    <DL>
      <DT>问题 1
        <DD>问题 1 的内容...
      <DT>问题 2
        <DD>问题 2 的内容...
      <DT><FONT color=green>问题 3</FONT>
        <DD>问题 3 的内容...
      <DT><FONT color=red>问题 4</FONT>
        <DD>问题 4 的内容...
    </DL>
  </BODY>
</HTML>
```

运行后的浏览器显示如图 3-27 所示，从图中可以看出定义列表的使用效果。

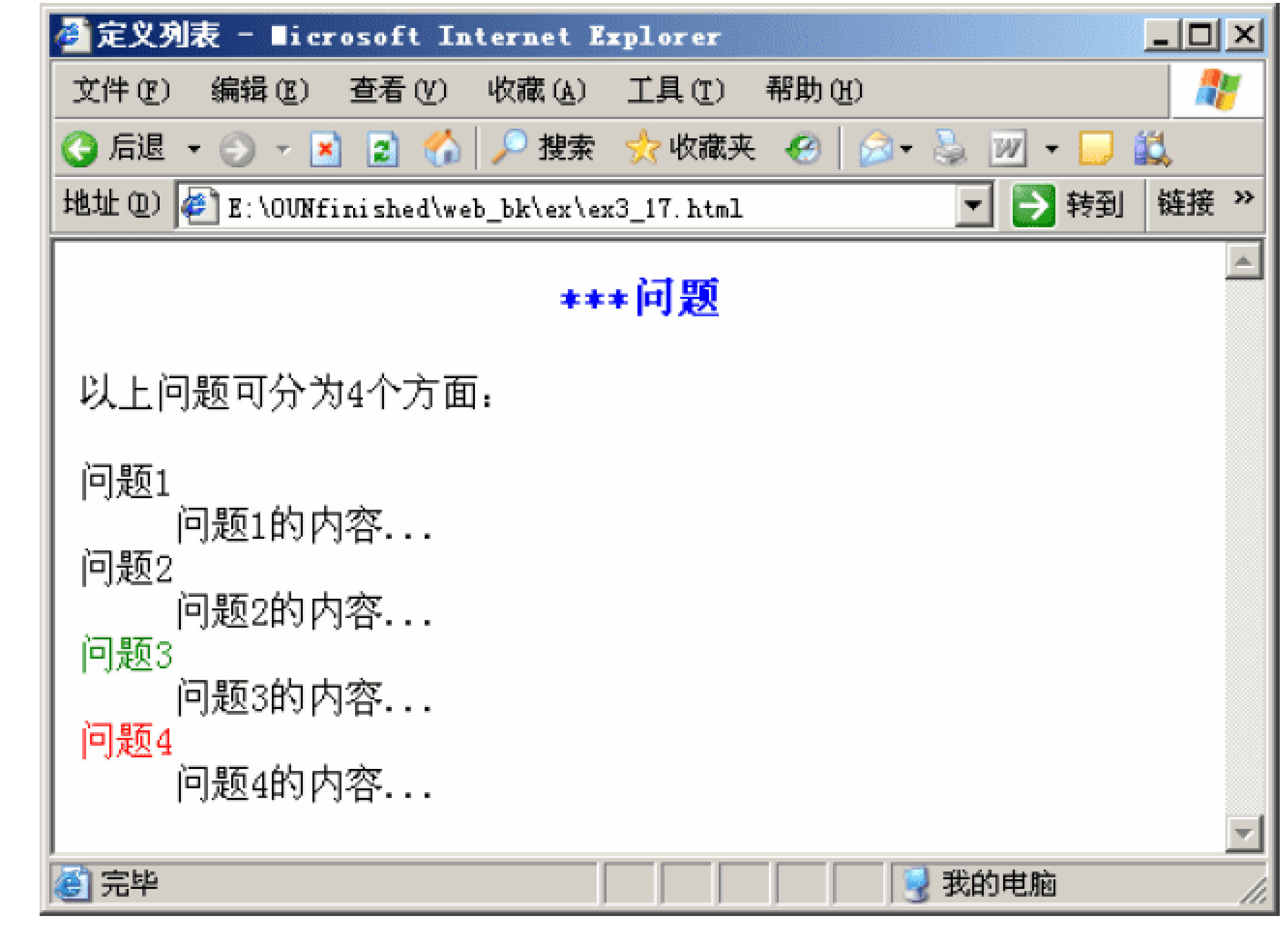


图 3-27　定义列表的应用

4. 列表的嵌套

有序列表和无序列表不仅可以自身嵌套，而且彼此可互相嵌套。列表嵌套把主页分为多个层次，例如书的目录，给人以很强的层次感。

【实例 3-18】有序列表中嵌套有序列表的应用

程序代码如 ex3_18.html 所示。

ex3_18.html

```
<HTML>
  <HEAD><TITLE>有序列表中嵌套有序列表</TITLE></HEAD>
  <BODY>
    <FONT color=blue size=3>有序列表中嵌套有序列表示例</FONT>
    <OL type="1">
      <LI>有序 1
      <OL type="a">
        <LI>有序 a
        <LI>有序 b
      </OL>
      <LI>有序 2
      <OL type="I">
        <LI>有序罗马 1
        <LI>有序罗马 2
        <LI>有序罗马 3
      </OL>
    </OL>
  </BODY>
</HTML>
```

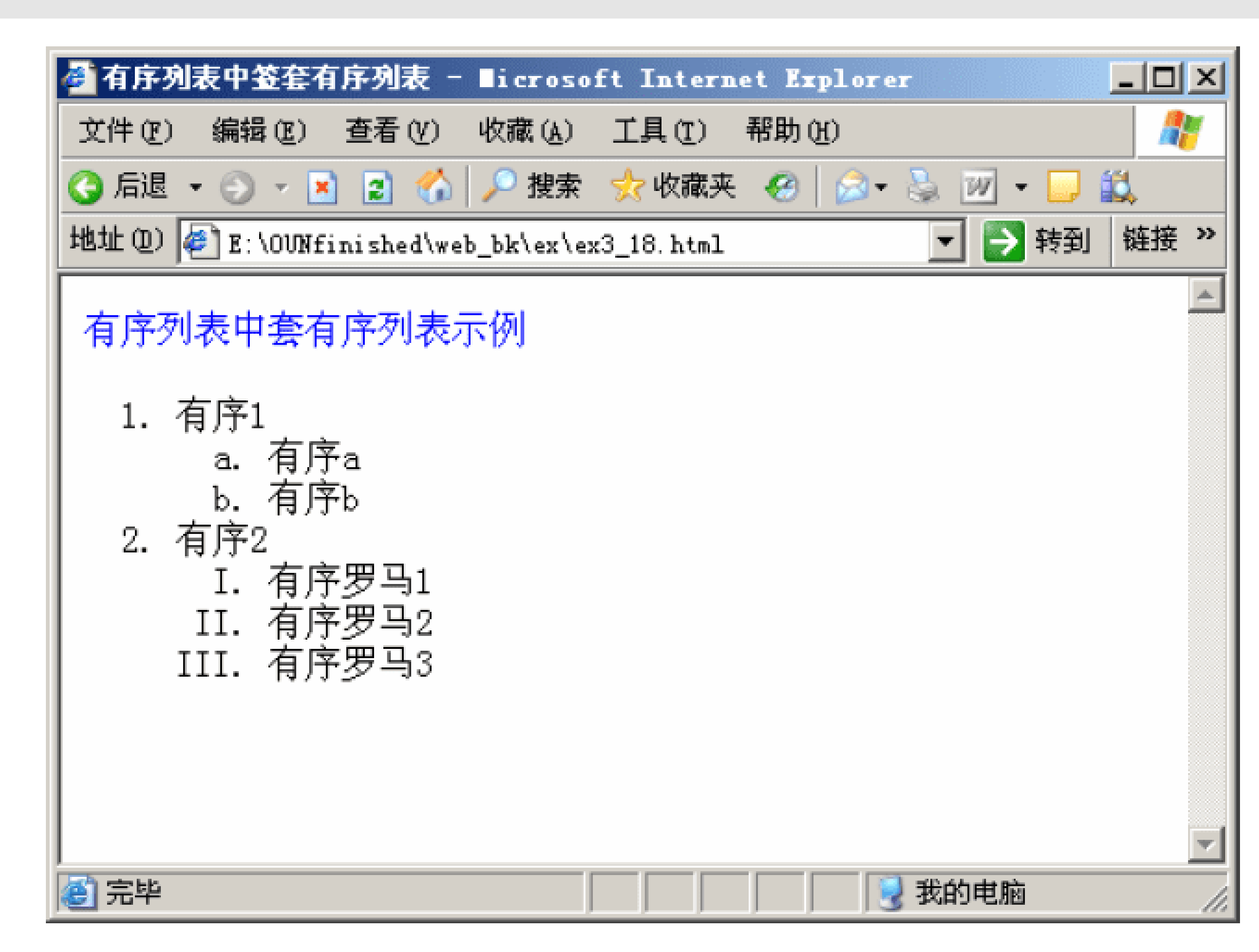


图 3-28　有序列表中嵌套有序列表的应用

运行后的浏览器显示如图 3-28 所示，从图中可以看出有序列表嵌套的使用效果。

【实例 3-19】有序列表中嵌套无序列表

程序代码如 ex3_19.html 所示。

ex3_19.html

```
<HTML>
  <HEAD><TITLE>有序列表中嵌套无序列表</TITLE></HEAD>
  <BODY>
    <FONT color=blue size=3>有序列表中嵌套无序列表示例</FONT>
    <OL type="1">
      <LI>有序 1
      <UL>
        <LI>无序
        <LI>无序
      </UL>
      <LI>有序 2
      <UL>
        <LI>无序
        <LI>无序
        <LI>无序
      </UL>
      <LI>有序 3
    </OL>
  </BODY>
</HTML>
```

运行后的浏览器如图 3-29 所示，图中可以看出有序列表中嵌套无序列表的使用效果。

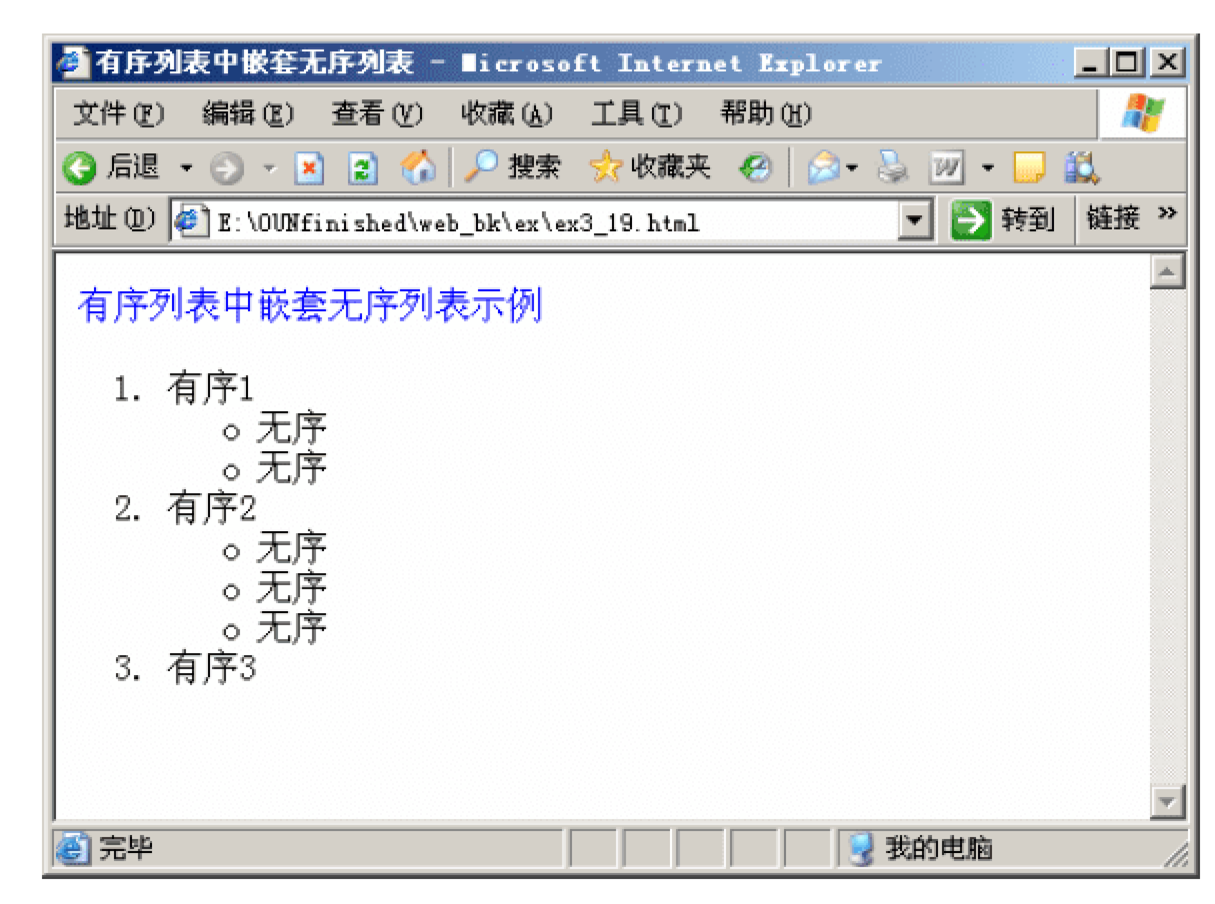


图 3-29　有序列表中嵌套无序列表的用法

3.3.4　超级链接

超链接是网页互相联系的桥梁，超链接(Hyperlink)可以看作是一个“热点”，它可以从当前 Web 页定义的位置跳转到其他位置，包括当前页的某个位置、Internet 或本地硬盘或局域网上的其他文件，甚至跳转到声音、图片等多媒体文件。利用超链接来浏览 Web 页是一种最普遍的应用，通过超链接还可以获得不同形态的服务，如文件传输、资料查询、电子邮件、远程访问等。

当 Web 页包含超链接时，Web 页中默认的外观形式为蓝色且带下划线的文字，或者直接使用图片。使用鼠标单击这些文本或图片，可跳转到相应位置。鼠标指针指向具有超链接的文本或图片时，将变成手形。

1. 锚点标签<A>

锚点(anchor)标签由<A>定义，它在网页上建立超文本链接。通过单击一个词、句或图片，可从此处转到另一个链接资源(目标资源)，这个目标资源有唯一的地址(URL)。具有以上特点的词、句或图片就称为热点。定义<A>标签的格式为：

```
<A href="链接的目标地址" name="链接名称字符串" target="打开窗口方式">浏览器中显示的热点
</A>
```

属性 href 为超文本引用，它的值为一个 URL，是目标资源的有效地址。在书写 URL 时要注意，如果资源放在自己的服务器上，可以使用相对路径。否则，应写绝对路径。href 不能与 name 同时使用。

属性name指定当前文档中的一个字符串作为链接时可以使用有效的目标资源的地址。

属性 target 设定链接被按后结果为所要显示的窗口。可选值为_blank、_parent、_self、_top、框架名称。属性 target 的定义及其对应的含义如表 3-6 所示。

表 3-6　超链接属性 target 的定义

target 的定义	含　义
target="_blank"	将链接的画面内容，显示在新的浏览器窗口中
target="new"	
target="_parent"	将链接的画面内容，显示在直接父框架窗口中
target="_self"	将链接的画面内容，显示在当前窗口中(默认值)
target="_top"	将框架中连接的画面内容，显示在没有框架的窗口中(除去了框架)
target="框架名称"	只运用于框架中，若被设定则链接结果将显示于该“框架名称”指定的框架窗口中，框架窗口名称是事先由框架标签所命名的

注意：

热点允许根据需要设置颜色，可利用<BODY>标签中相关的属性。

2. 创建指向其他页面的链接

创建指向其他页面的链接，就是在当前页面与其他相关页面间建立超链接。无论目标文件与当前文件的目录关系如何，其格式为：

```
<A href="目标地址的路径/目标文件名.html">热点文本</A>
```

根据目标文件与当前文件的目录关系，有以下几种写法。

(1) 链接到同一目录内的网页文件(使用相对地址)

链接到同一目录内的网页文件的格式为：

```
<A href="目标文件名.html">浏览器中的热点文本</A>
```

其中的目标文件名是链接所指向的文件。

【实例 3-20】链接到同一目录内的网页文件

以下的代码演示了使用相对路径的链接，程序代码如 ex3_20.html 所示。

ex3_20.html

```
<HTML>
  <HEAD>
    <TITLE>我的主页</TITLE>
  </HEAD>
  <BODY>
    <H2 align=center><FONT color=green>欢迎来到我的主页</FONT></H2>
    <CENTER>
      <A href="new.html">最新更新</A><BR>
      <A href="self.html">我的自传</A><BR>
      <A href="message.html" target="_blank">给我留言</A><BR> <!--新的浏览器窗口显示-->
    </CENTER>
  </BODY>
</HTML>
```

运行后的浏览器显示如图 3-30 所示，从图中只能看到超链接，当把鼠标移动到超链接上时，鼠标指针变为手形，单击链接并打开指定的网页(new.html、self.html 及 message.html，文件需要另外单独建立)。如果在<A>标签中省略属性 target，则在当前窗口中显示；当 target="_blank"时，将在新的浏览器窗口中显示，因此单击第三个超链接后可以看到相关内容在一个新窗

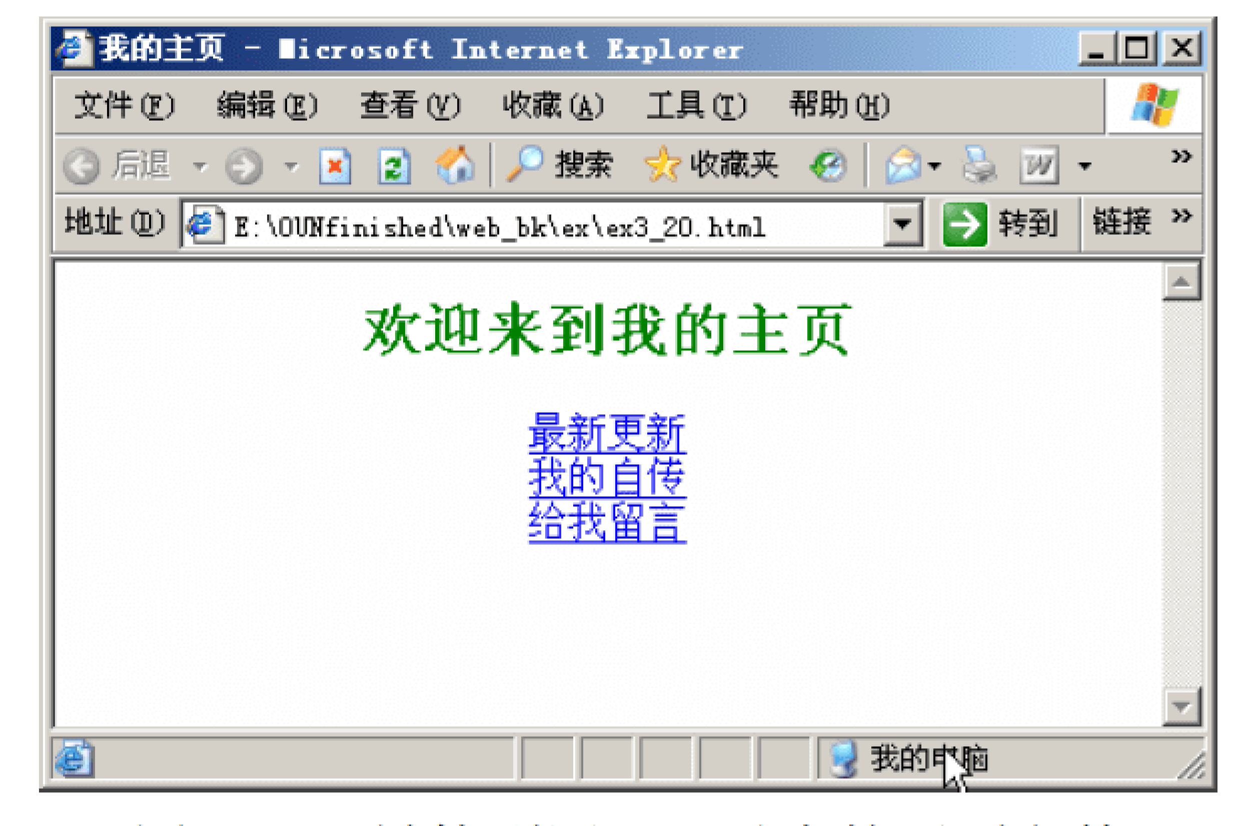


图 3-30　链接到同一目录内的网页文件

口中显示。

(2) 链接到本站点下不同目录中的网页文件(使用相对地址)

链接到下一级目录中网页文件的格式为：

```
<A  href="子目录名/目标文件名.html">热点文本</A>
```

链接到上一级目录中网页文件的格式为：

```
<A  href=". . /目标文件名.html">热点文本</A>
```

其中“. . /”表示退到上一级目录中。读者可以根据这个基本原则建立本网站下各个页面之间的链接。

(3) 链接到其他网站的网页文件(使用绝对地址)

链接到其他网站下的网页文件的格式为：

```
<A  href="网络协议://网站名称/子目录名/目标文件名.html">热点文本</A>
```

使用这种方法，可以建立不同网站间的网状关系，这个特性是 Web 技术的一个基本特征之一。

URL 的基本语法为，网络协议://URL 地址[:PORT]/PATH/FILE。其中网络协议可以是 File、HTTP、Gopher、WAIS、News 及 Telnet 等。

3. 创建指向本页中的链接

要在当前页面实现超链接，需要定义两个标签，一个为超链接标签，另一个为书签标签。超链接标签的格式为：

```
<A  href="#书签名">热点文本  </A>
```

即单击热点文本，将跳转到“记号名”开始的文本。

书签就是用<A>标签对该文本作一个记号。如果有多个链接，不同目标文本要设置不同的书签名，书签名在<A>的 name 属性中定义。格式为：

```
<A  name="书签名">目标文本附近的字符串</A>
```

【实例 3-21】指向本网页中的链接

程序代码如 ex3_21.html 所示。

ex3_21.html

```
<HTML>
  <HEAD><TITLE>我的主页</TITLE></HEAD>
  <BODY link=red alink=blue vlimk=green>
    <A name="main"> </A>
    <H2 align=center><B>欢迎来到我的主页</B></H2>
    <FONT size=3 color=purple>
      <CENTER>
```

```
            <A href="#new">最新更新</A>    
            <A href="#self">我的自传</A>    
            <A href="#message">给我留言</A>       <P>
        </CENTER>
    </FONT>
    <A name="new"></A>
        最新更新<BR>
        ....<BR>....<BR>....<BR>....<BR>
        <DIV align=center><A href="#main">返回</A></DIV><P>
    <A name="self"></A>
        我的自传<BR>
        ....<BR>....<BR>....<BR>....<BR>
        <DIV align=center><A href="#main">返回</A></DIV><P>
    <A name="message"></A>
        给我留言<BR>
        ....<BR>....<BR>....<BR>....<BR>
        <DIV align=center><A href="#main">返回</A></DIV><P>
  </BODY>
</HTML>
```

运行后的浏览器显示如图 3-31 所示，由于本实例为页内链接及网页的垂直高度有限，因此浏览器的高度不宜过大，否则看不到效果。单击左图三个链接之一，浏览器会转向相应的位置。

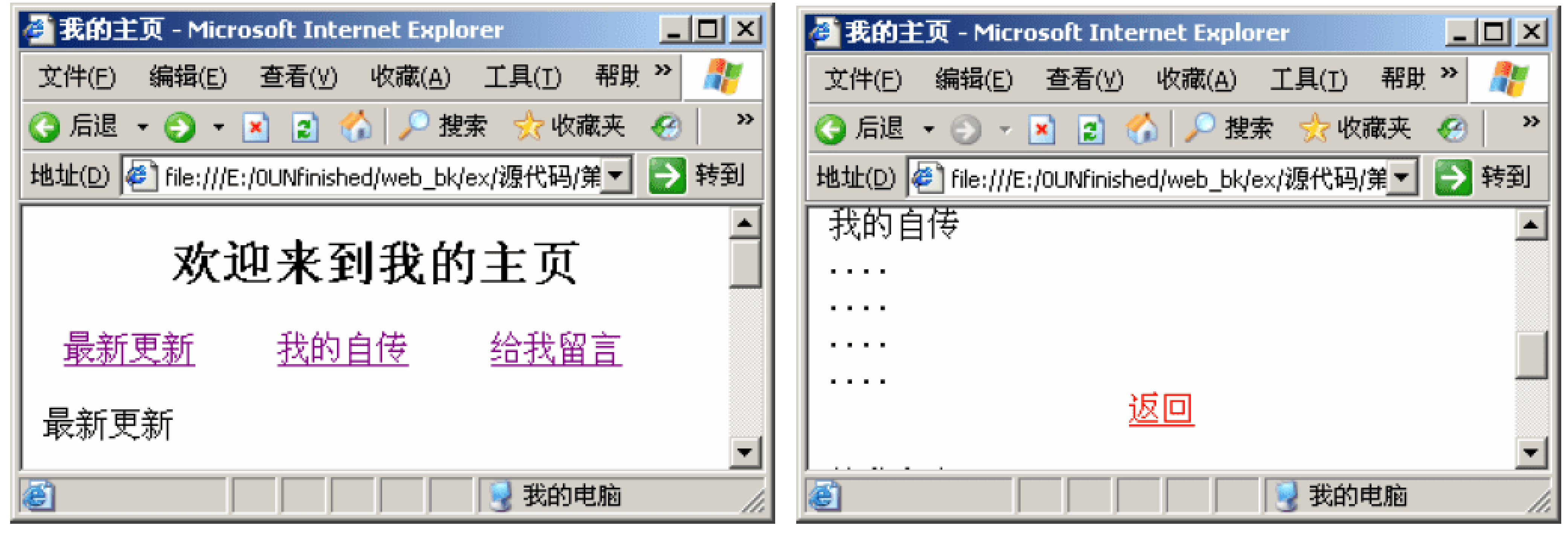


图 3-31　指向本网页中的链接

由上例可以看出，指向本页面其他部分的链接与指向别的页面的链接不同之处，在于后者要在页面中定义标签。两种链接的相同之处在于它们都使用了<A>标签。

4. 创建指向其他类型文件的链接

如果链接到的文件不是 HTML 文件，则浏览器会提示对该文件如何操作，如将该文件作为下载文件或打开等。创建这种类型链接的方法为：

```
<A   href="下载文件名">热点文本</A>
```

文件可以放在本网站内，也可以是位于其他网站的，在书写 href 属性时可以参考前面提到的用法，而文件的类型不仅局限于 html 类型，也可以是其他类型。

5. 创建发送电子邮件的链接

单击指向电子邮件的链接，将打开默认的电子邮件程序，如 FoxMail、Outlook Express，

并自动填写收件人地址。要创建指向电子函件的链接，可以在<A>标签的 href 属性中加入 mailto，其格式为：

```
<A   href="mailto:E-mail 地址">发送邮件的热点文本</A>
```

【实例 3-22】创建发送电子邮件的链接

程序代码如 ex3_22.html 所示。

ex3_22.html

```
<HTML>
  <HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=GB2312">
    <TITLE>使用超链接来传送电子邮件</TITLE>
  </HEAD>
  <BODY>
    <P>请将对此网页内容的意见或感想，
      <A HREF="mailto: author@163.com">发送邮件给我</A>
      ，谢谢！
    </P>
  </BODY>
</HTML>
```

运行后的浏览器显示如图 3-32 中左图所示，从图中只能看到一个“发送邮件给我”的超链接，当把鼠标指针移到超链接上时，鼠标指针变为手型，单击则打开了本机发送电子邮件的应用程序 Outlook Express；如图 3-32 中右图所示，其中收件人的地址已经自动填写了。

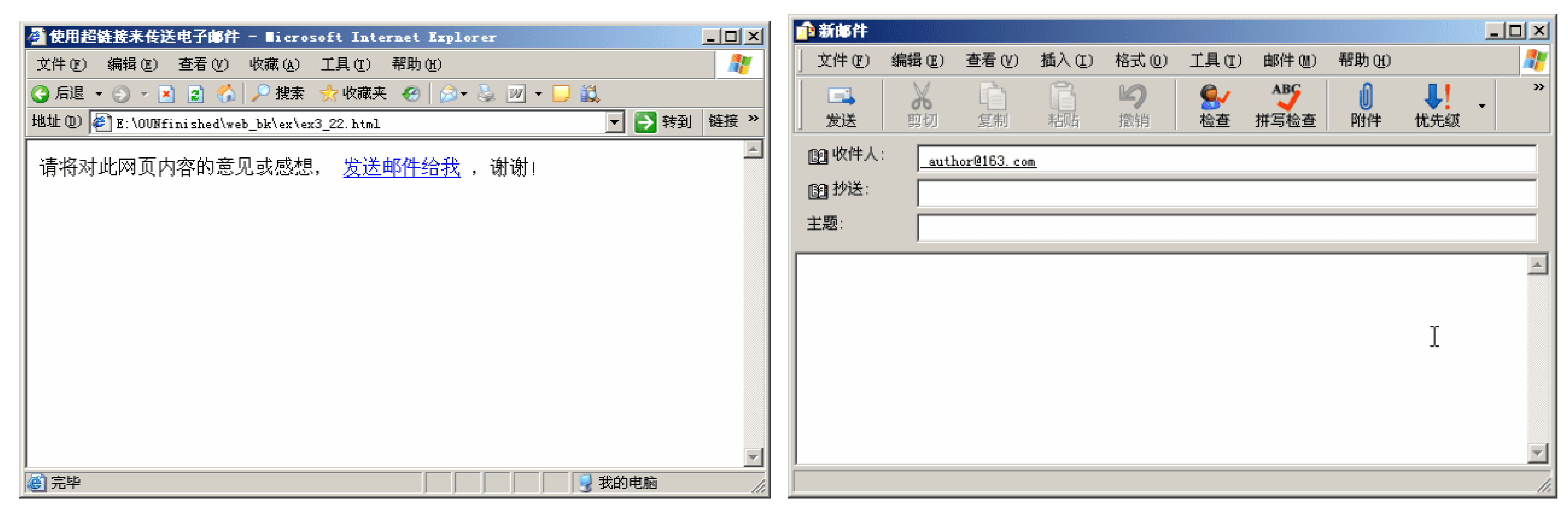


图 3-32　创建发送电子邮件的链接

3.3.5　表格

HTML提供了<TABLE>元素，这使得网页创作者可以创建表格，表格可将文本和图片等素材按行、列排列。与列表一样，表格有利于排版，它常用来控制显示格式，使整个页面更规则地放置不同的素材，使得条目排列清晰。

1. 建立简单表格

最简单的表格仅包括行和列。表格的标签为<TABLE>，行的标签为<TR>，表项的标签为<TD>。其中，<TR>是单标签，一行的结束是新一行的开始。表项内容写在<TD>与

</TD>之间。<TABLE>则必须成对使用。格式为：

```
<TABLE border=n width=x 或 x% height=y 或 y% cellspacing=i cellpadding=j>
  <TR> <TH>表头 1</TH><TH>表头 2</TH><TH>…</TH><TH>表头 n</TH>
  <TR> <TD>表项 1</TD><TD>表项 2</TD><TD>…</TD><TD>表项 n</TD>
    …
  <TR> <TD>表项 1</TD><TD>表项 2</TD><TD>…</TD><TD>表项 n</TD>
</TABLE>
```

由上面格式可以看出，表格是一行一行建立的，在每一行中填入该行每一列的表格数据项。把表头看作一行，只不过用<TH>标签。

格式定义中使用的标签及其含义如下。

- <TABLE>：定义一个表格；一般它不直接包含 TR、TH/TD 元素，而是包含 0 个或 1 个 CAPTION 元素(标题)、若干个 COL/COLGROUP 元素(列组)、0 个或 1 个 THEAD/TFOOT 元素(表头/表尾)、1 个以上的 TBODY 元素(表体)。
- <TR>：Table Row，它定义表格中的一行；每个 TR 元素可以包含 1 个以上的 TH 元素或者 TD 元素。
- <TH>：Table Head，用于定义表头信息；可以包含任何类型的 HTML 元素。
- <TD>：Table Data，用于生成数据单元；可以包含任何类型的 HTML 元素。

在浏览器中显示时，<TH>标签的文字按粗体显示，<TD>标签的文字按正常字体显示。表格的整体外观由<TABLE>标签的以下属性决定。

- border：定义表格边框的粗细，n 取整数，单位是像素。如果省略，则不带边框。
- width：定义表格的宽度，x 为像素数或占窗口的百分比。
- height：定义表格的高度，y 为像素数或占窗口的百分比。
- cellspacing：定义表项间隙，单位是像素。
- cellpadding：定义表项内部空白，单位是像素。

也可在第一列加表头，其格式为：

```
<TABLE border=n width=x 或 x% height=y 或 y% cellspacing=i cellpadding=j>
  <TR> <TH>表头 1</TH><TD>表项 1</TD><TD>…</TD><TD>表项 n</TD>
  <TR> <TH>表头 2</TH><TD>表项 1</TD><TD>…</TD><TD>表项 n</TD>
    …
  <TR> <TH>表头 n</TH><TD>表项 1</TD><TD>…</TD><TD>表项 n</TD>
</TABLE>
```

对于表格的宽度，如果为像素数，则为绝对宽度。如果为百分数%，则为相对于浏览器窗口宽度的比例，也就是100%时表格宽度等于窗口的宽度。不管是绝对宽度还是相对宽度，数值太小不足以显示表格中的内容时，会自动以最小的宽度显示。

【实例 3-23】带标题的表格

程序代码如 ex3_23.html 所示。

ex3_23.html

```
<HTML>
  <HEAD>
      <TITLE>销售业绩</TITLE>
  </HEAD>
  <BODY>
    <TABLE border>
    <CAPTION>张三销售业绩</CAPTION>
    <TR><TH>编号</TH><TH>姓名</TH><TH>外销</TD><TH>内销</TH><TH>总数</TH>
    <TR><TD>0001</TD><TD>张　三</TD><TD>45</TD><TD>86</TD><TD>131</TD>
   </TABLE>
   <BR>
   <TABLE border>
    <CAPTION align=right>张三销售业绩</CAPTION>
    <TR><TH>编号</TH><TH>姓名</TH><TH>外销</TD><TH>内销</TH><TH>总数</TH>
    <TR><TD>0001</TD><TD>张　三</TD><TD>45</TD><TD>86</TD><TD>131</TD>
   </TABLE>
   <BR>
   <TABLE border>
    <CAPTION align=left valign=bottom>张三销售业绩</CAPTION>
    <TR><TH>编号</TH><TH>姓名</TH><TH>外销</TD><TH>内销</TH><TH>总数</TH>
    <TR><TD>0001</TD><TD>张　三</TD><TD>45</TD><TD>86</TD><TD>131</TD>
   </TABLE>
   <BR>
  </BODY>
</HTML>
```

运行后的浏览器显示如图 3-33 所示，从图中可以看到三种不同格式的表头。第一个表格的表头为居中对齐，而第二个和第三个分别为右对齐和左对齐。

由此可见，用<CAPTION>标签给表格加标题，其格式为：

```
  <TABLE>
    <CAPTION align=left|right|top|bottom
valign=top|bottom>标题</CAPTION>
  </TABLE>
```



图 3-33　带标题的表格

其中，各元素所表示的含义如下。

valign：设置标题放在表的上部 top(默认)，还是下部 bottom。

align：设置标题放在表的上部或下部的中间 center(默认)、左边 left，还是右边 right。也可设置标题放在表的上部 top(默认)，还是下部 bottom。

注意：

还可以用文字标签对标题或表格中的文字进行修饰，如设置字的大小、字体、颜色等。

2. 跨多行、多列的表项

使用<TR>、<TD>、<TH>标签的 colspan 和 rowspan 属性，可以分别制作跨多行(合并行)和跨多列(合并列)的表格。

(1) 跨多列表项

跨多列表项的格式为：

```
<TD  colspan=x> 表项 </TD>
```

或

```
<TR  colspan=x> 表项 </TR>
```

或

```
<TH  colspan=x> 表项 </TH>
```

其中，x 表示合并的列数。

(2) 跨多行表项

跨多行表项的格式为：

```
<TD  rowspan=y> 表项 </TD>
```

或

```
<TR  rowspan=y> 表项 </TR>
```

或

```
<TH  rowspan=y> 表项 </TH>
```

其中，y 表示合并的行数。

(3) 同时跨多列多行表项

在<TH>中同时使用 colspan 和 rowspan 属性可制作多重表头。格式为：

```
<TH  rowspan=x  colspan=y>
```

其中，rowspan 设置表头跨过 x 列，colspan 设置表头跨过 y 行。

【实例 3-24】 跨多行、多列的表项

程序代码如 ex3_24.html 所示。

ex3_24.html

```
<HTML>
  <BODY>
    <TABLE border>
      <TR>  <TH colspan="3">B040801 班</TH>
      <TR>  <TH>姓名</TH><TH>性别</TH><TH>年龄</TH>
      <TR>  <TD>张三</TD><TD>男</TD><TD>21</TD>
```

```
        <TR>   <TD>王二</TD><TD>女</TD><TD>22</TD>
      </TABLE>
      <BR>
      <TABLE border align="center">
       <TR> <TH rowspan="3">B040801 班</TH> <TH>姓名</TH> <TD>性别</TD><TD>年龄</TD>
        <TR>   <TD>张三</TD><TD>男</TD><TD>21</TD>
        <TR>   <TD>王二</TD><TD>女</TD><TD>22</TD>
      </TABLE>
    </BODY>
  </HTML>
```



图 3-34　跨多行、多列的表项

运行后的浏览器显示如图 3-34 所示，从图中可以看到跨多行、多列的表项。

【实例 3-25】创建多重表头的表格

程序代码如 ex3_25.html 所示。

ex3_25.html

```
  <HTML>
    <HEAD><TITLE>学生成绩</TITLE></HEAD>
    <BODY>
      <TABLE border=3>
        <CAPTION><FONT size=6><B>学生成绩表</B></FONT></CAPTION>
        <TR> <TH rowspan=2>学号<TH rowspan=2>姓名<TH colspan=3>成绩
        <TR>   <TH>Java<TH>网页制作<TH>数据库
        <TR> <TD>0001<TD>张三<TD>92<TD>69<TD>161
        <TR> <TD>0002<TD>王五<TD>86<TD>92<TD>178
        <TR> <TD>0003<TD>李四<TD>90<TD>100<TD>190
        <TR> <TD>0004<TD>何六<TD>72<TD>86<TD>158
        <TR> <TD>0005<TD>赵七<TD>80<TD>93<TD>173
      </TABLE>
    </BODY>
  </HTML>
```

运行后的浏览器显示如图 3-35 所示，其中的多重表头表格就是将 rowspan 和 colspan 结合后实现的。



图 3-35　创建多重表头的表格

3. 表格在页面中的属性

前面介绍的是表格中的各个单元格的属性的设定。现在，把表格作为一个整体，介绍表格在页面中的位置及背景的设置。

(1) 设定表格在页面中的位置

与图片一样，表格在浏览器窗口中的位置也有三种：居左、居中和居右。这个设置使用<TABLE>标签的 align 属性，其格式为：

```
<TABLE  align=left|center|right>
```

其中，left 表示居左，center 表示居中，right 表示居右。当表格位于页面的左侧或右侧时，文本填充在另一侧；当表格居中时，表格两边没有文本；当 align 属性默认时，文本在表格的下面。对此，读者可以参考【实例 3-24】，其第一个表格未设置表格对齐，因此默认的为左对齐，而第二个表格使用了表格居中对齐的功能。

(2) 表格的颜色和图片背景

由于表格能够方便地控制文字、图片的位置，所以很多网页常用表格来制作。下面介绍利用表格在这方面的用法。首先，在<TABLE>、<TH>、<TR>、<TD>标签中，均可以使用下面的属性。

- background="图片文件名"：设置背景图片；
- bordercolor="颜色"：设定表格边框的颜色；
- bordercolorlight="颜色"：设定表格边框亮部的颜色；
- rules="row,cols 或 none"：设定表内线的显示方法。

注意：

还可以用<TD>、<TH>标签中的 bgcolor 属性给表格的每个单元设置背景色，如<TD bgcolor="颜色或颜色值">。

【实例 3-26】创建带有背景图片的表格

程序代码如 ex3_26.html 所示。

ex3_26.html

```
<HTML>
  <HEAD><TITLE>学生成绩</TITLE></HEAD>
  <BODY>
    <TABLE border=3 align=center background="backgr1.jpg" width=500 height=300>
      <CAPTION><FONT size=6><B>学生成绩表</B></FONT></CAPTION>
      <TR> <TH rowspan=2>学号<TH rowspan=2>姓名<TH colspan=3>成绩
      <TR>  <TH>Java<TH>网页制作<TH>数据库
      <TR> <TD>0001<TD>张三<TD>92<TD>69<TD>161
      <TR> <TD>0002<TD>王五<TD>86<TD>92<TD>178
      <TR> <TD>0003<TD>李四<TD>90<TD>100<TD>190
      <TR> <TD>0004<TD>何六<TD>72<TD>86<TD>158
      <TR> <TD>0005<TD>赵七<TD>80<TD>93<TD>173
    </TABLE>
  </BODY>
</HTML>
```

运行后的浏览器显示如图 3-36 所示，图中的表格具有背景图片。

如果将表格中<TABLE>元素的 border 属性设置为 0，则可以生成无边框的表格，这种表格通常可用于排版。

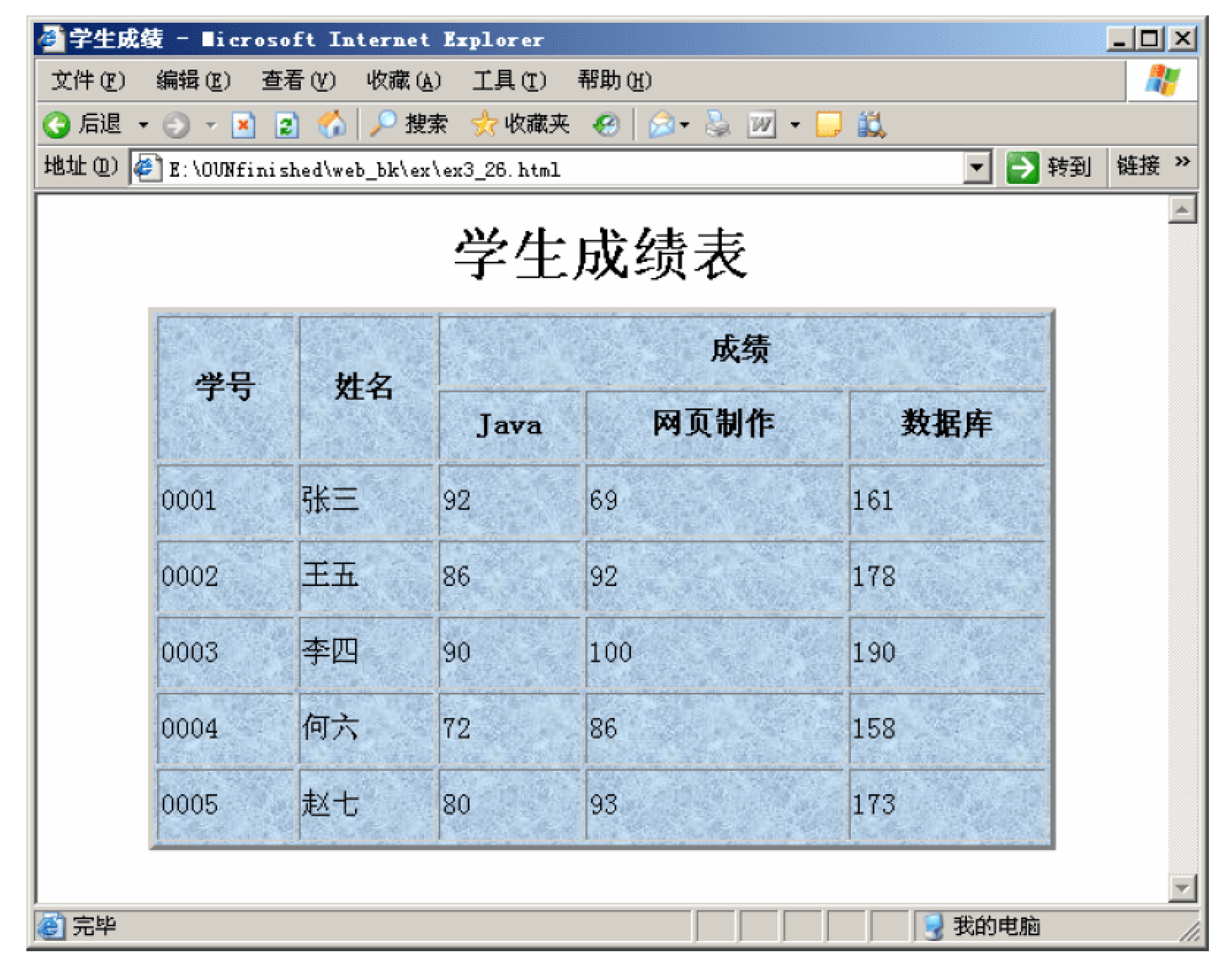


图 3-36　创建带有背景图片的表格

4. 表格的分组显示

复杂表格指的是对表格的行、列的对齐方式进行设置。

(1) 按行分组

表格的行从上到下可以分为表头、表体和表尾。分别使用标签<THEAD>、<TBODY>、<TFOOT>来定义。这种定义方法不但可以实现设置表头，而且可以将表格的行分组，其格式为：

```
<TABLE>
  <THEAD> 题头 </THEAD>
  <TFOOT> 表尾 </TFOOT>
  <TBODY> 表格主体 1 </TBODY>
      …
  <TBODY> 表格主体 n </TBODY>
</TABLE>
```

可以定义多个表体<TBODY>部分，每个<TBODY>部分定义多行信息，每行的格式与一般表格中的一样，在同一个<TBODY>中，所有各行的列数必须相同。<THEAD>、<TBODY>、<TFOOT>标签可以是单标签。

在浏览器中显示时，表头、表尾以及各个表体之间都用边框来分隔。

【实例 3-27】按行分组制作表格

程序代码如 ex3_27.html 所示。

ex3_27.html

```
<HTML>
  <BODY>
    <TABLE border align=center>
      <THEAD>
```

```
      <TR bgcolor=pink>  <TH>学号<TH>姓名<TH>性别<TH>年龄
    </THEAD>
    <TBODY>
      <TR>
        <TD bgcolor=orange>0001</TD>
        <TD bgcolor=greenyellow>王二</TD>
        <TD bgcolor=cyan>女</TD>
        <TD bgcolor=red>21</TD>
      </TR>
      <TR>
        <TD bgcolor=orange>0002</TD>
        <TD bgcolor=greenyellow>张三</TD>
        <TD bgcolor=cyan>男</TD>
        <TD bgcolor=red>20</TD>
      </TR>
    </TBODY>
  </TABLE>
 </BODY>
</HTML>
```

运行后的浏览器显示如图 3-37 所示，图中的表格就是一个按行分组方式建立的表格。

注意：

【实例 3-27】及后面的部分实例中所显示的表格具有不同的背景色，但受到色彩的限制，这里的图不能完全展示实际的效果，请读者自行运行实例代码以观察真实的显示效果。



图 3-37　按行分组方式建立的表格

(2) 按列分组

用列组<COLGROUP>标签和列<COL>标签，可以改变列的一些性质，如列的宽度等。列组可以包括一个或多个列，使用<COLGROUP>标签可一次设定列组中的列数以及各列的属性，其格式为：

```
<COLGROUP  span=x  width=y>
```

其中，span 的值 x 大于 0，表示从左数起，指定属性列组的列数，默认为 1 列(如 span=3，表示从第 1 列到第 3 列共 3 列)。width 的值 y 表示该列组各列的宽度，可以为像素数、%数(占当前浏览器窗口的百分比)或 n*(自动分配，为当前列的 1/n)。

<COL>标签可以设定一列的属性，它可以放在<COLGROUP>中使用，也可单独用于定义列组以外列的属性，但它不能构造列组，其格式为：

```
<COL  span=x  width=y>
```

其中，span 取 0 以上的值，用于指定所含列数，指本列及后续 x-1 列。width 的值是列的宽度，宽度属性值的取法同<COLGROUP>。

【实例 3-28】按列分组制作表格，设置列宽

程序代码如 ex3_28.html 所示。

ex3_28.html

```
<HTML>
  <HEAD><TITLE>学生</TITLE></HEAD>
  <BODY>
    <TABLE border width=250 align=center>
      <CAPTION>学生名单</CAPTION>
      <COLGROUP span=2 width=90>
      <THEAD>
        <TR> <TH>姓名<TH>性别<TH>年龄
      </THEAD>
        <TR>
          <TD bgcolor=greenyellow>王二</TD>
          <TD bgcolor=cyan>女</TD>
          <TD bgcolor=red>21</TD>
        </TR>
        <TR>
          <TD bgcolor=greenyellow>张三</TD>
          <TD bgcolor=cyan>男</TD>
          <TD bgcolor=red>20</TD>
        </TR>
    </TABLE>
  </BODY>
</HTML>
```

运行后的浏览器显示如图 3-38 所示，图中的表格是按列分组的并设置了不同的列宽。

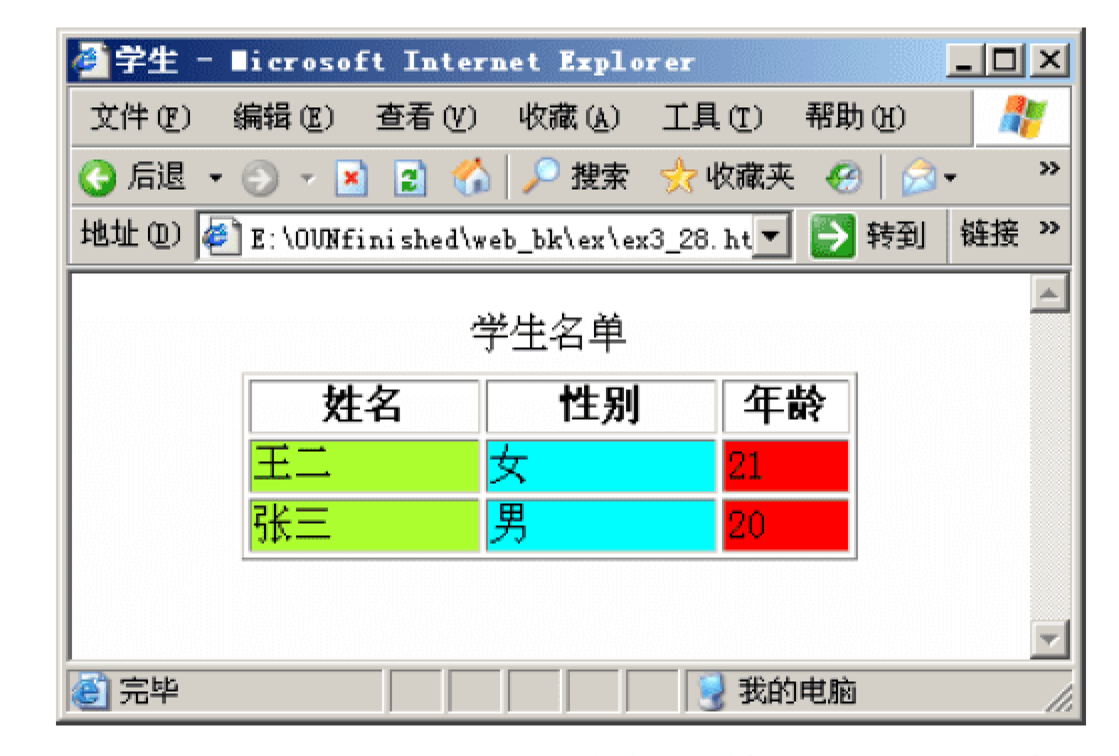


图 3-38　按列分组制作表格，设置列宽

5. 对齐表项

在默认设置下，表项居于单元格的左端。可用列、行的属性来设置表项数据在单元格中的水平和垂直对齐。

(1) 水平对齐

水平数据分为：表项数据的居中、左对齐、右对齐、左右调整、整列以特殊文字对齐。方式是用标签<COL>、<COLGROUP>、<TH>、<TD>和<TR>的 align 属性。align 的属性值分别如下。

- center：居中；
- left：左对齐；
- right：右对齐；
- justify：左右调整。

【实例 3-29】表格的水平对齐

表格可以使用不同的水平对齐方式，程序代码如 ex3_29.html 所示。

ex3_29.html

```
<HTML>
  <HEAD><TITLE>学生名单</TITLE></HEAD>
  <BODY>
    <TABLE border=3 align=center width=60%>
      <COLGROUP align=left>
      <COLGROUP align=center>
      <COLGROUP align=right>
      <COLGROUP align=jusitfy>
      <CAPTION>学生名单表</CAPTION>
      <TR>   <TH>学号<TH>姓名<TH>性别<TH>年龄
      <TR>   <TD>0001<TD>李 四<TD>女<TD>21
      <TR>   <TD>0002<TD>张 三<TD>男<TD>20
    </TABLE>
  </BODY>
</HTML>
```

运行后的浏览器显示如图 3-39 所示，其中的表格使用了不同的水平对齐方式，请读者仔细对照表头和表体，找出各种水平对齐方式之间的差别。

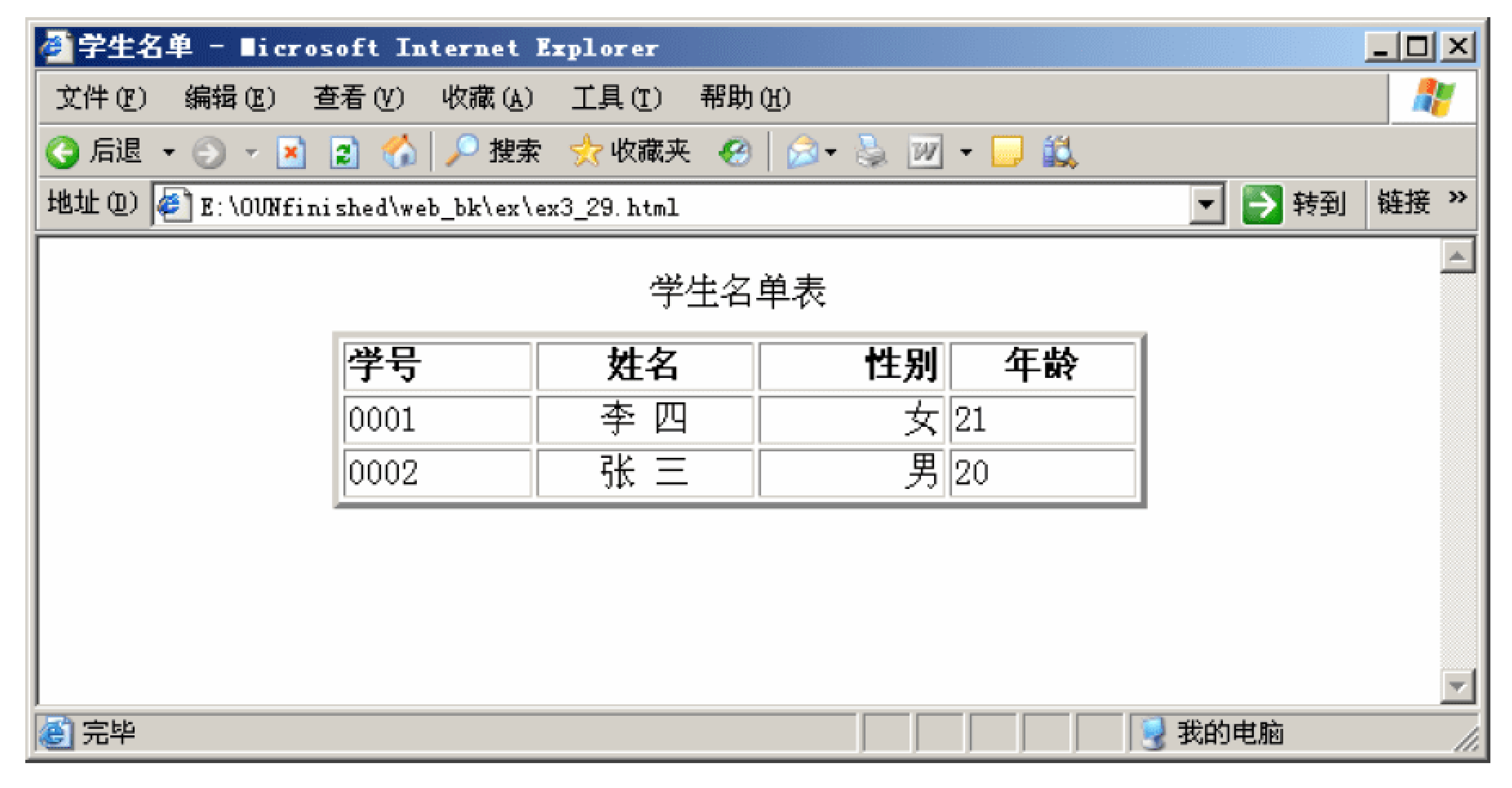


图 3-39　表格的水平对齐

(2) 垂直对齐

垂直对齐由 valign 属性设定。其中 valign 的属性值分别如下。

- top：数据项靠单元格顶；
- bottom：数据项靠单元格底；
- middle：数据项靠单元格中；
- baseline：同行单元数据项位置一致。

与前面介绍的水平对齐类似，请读者自行验证。

3.3.6　图像

图像是美化网页最常用的元素之一。目前，在网页中使用的图片，通常为 GIF 格式和 JPEG 格式。

GIF 格式文件最多只能显示 256 种颜色，这使得它很少用于存储照片。但是，存放图标、剪贴画和艺术线条等对颜色要求不高的图片，已经足够了。GIF 格式图片的优点在于制作透明、隔行和动画效果。

JPEG 格式文件可以拥有计算机所能提供的最多种颜色，适合存放高质量的彩色图片、照片。另外，JPEG 格式文件采用压缩方式存储文件信息、相同的图片，所占空间比 GIF 文件小，所以下载时间较短，浏览速度较快。但是，JPEG 格式的文件没有 GIF 格式文件的三种特殊效果。

以下从网页背景、图片标签、图片布局等方面加以说明。

1. 设置网页的背景

网页的背景可以是某种颜色，也可以是图片。无论是图片，还是背景色，都使用<BODY>标签来说明，但需要使用不同的属性。

(1) 设置背景色

利用色彩做背景，比较容易在颜色上协调，而且下载速度比采用图片作为背景快。网页默认为白色，<BODY>的 bgcolor 属性用于设置网页的背景色，设置的格式为：

```
<BODY   bgcolor="颜色值">
```

(2) 使用背景图片

利用<BODY>标签的 background 属性，可为网页铺上背景图片，设置的格式为：

```
<BODY   background="图片文件名">
```

其中，background 取值为一个图片文件名，并且要指出文件存放的路径，可以是相对路径或绝对路径。图片文件可为 GIF 格式、JPEG 格式或其他兼容格式的文件。

在浏览器中，作为背景的图片将按原来的大小复制，重复铺满整个网页。作为背景的图片文件，可以做得很小，以便加快下载速度。

【实例 3-30】设置网页的背景图片

下面的例子为网页设置了背景图片，程序代码如 ex3_30.html 所示。

ex3_30.html

```
<HTML>
  <HEAD> <TITLE>图片背景</TITLE> </HEAD>
   <BODY background="backgr1.jpg">
   <H4 align=CENTER>用图片作为背景</H4>
   <FONT size=3>
    在浏览器中，作为背景的图片将按原来的大小复制，平铺而充满整个网页的显示区域。<BR>
    因此作为背景的图片文件，可以做得很小，这样可以加快下载的速度。<BR>
     </FONT>
   </BODY>
</HTML>
```

运行后的浏览器显示如图 3-40 所示，图中的网页显示区域内出现了背景图片，且较小

面积的背景图片经过平铺后充满了整个网页，文字显示在图片上方。

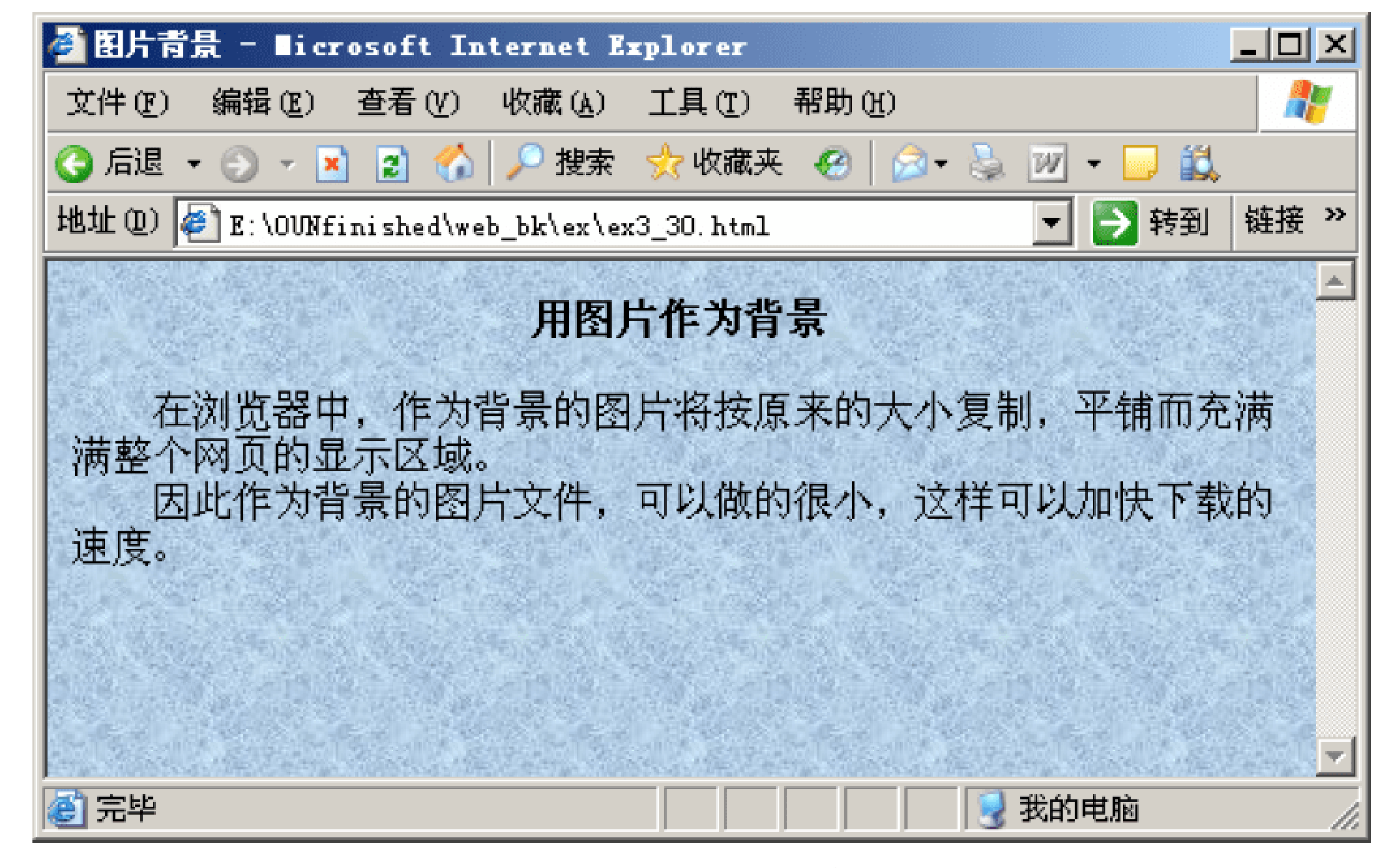


图 3-40　设置网页的背景图片

通常背景图片是经过特别制作的，它的上下以及左右的边被做成连续的，否则平铺后会出现断裂的效果。

2. 图片标签

使用图片标签，可以把一幅图片加入到网页中。用图片标签还可以设置图片的替代文本、尺寸、布局等属性，其格式为：

```
<IMG  src="图片文件名" alt="简单说明" width="图片的宽度" height="图片的高度" border="边框宽度" hspace="水平方向空白" vspace="垂直方向空白" align="对齐方式">
```

标签中的属性说明如表 3-7 所示。

表 3-7　IMG 标签的属性说明

属　性	说　明
src	指出要加入图片的文件名，即“图片文件的路径/图片文件名”
alt	在浏览器尚未完全读入图片时，在图片位置显示的文字
width	宽度(像素数或百分数)，通常只设为图片的真实大小以免失真，若需要改变图片大小最好事先使用图片编辑工具
height	设定图片的高度(像素数或百分数)
hspace	设定图片边沿空白，以免文字或其他图片过于贴近；设定图片左右的空间水平方向的空白像素数
vspace	设定图片上下的空间，空白高度采用像素作单位
align	图片在页面中的对齐/布局方式，或图片与文字的对齐方式

注意：

对于图片，可以使用<IMG>标签的 width 和 height 属性来设置图片的大小。其中 width 和 height 属性的值可取像素数，也可取百分数。如果不设定图片的尺寸，图片将按照其本身的大小显示。

【实例 3-31】在网页中添加图片

下面的例子在网页中添加图片，程序代码如 ex3_31.html 所示。

ex3_31.html

```
<HTML>
  <head>
    <title>在网页中添加图片</title>
  </head>
  <BODY>
    <img src="cartoon.gif">卡通 I </img>
    <br>
    <img src="cartoon.gif" alt=大卡通  width="90" height="20%">卡通 II </img>
  </BODY>
</HTML>
```

运行后的浏览器显示如图 3-41 所示，上面的小图为原图，下面的图片改变了图片的大小，读者可以改变浏览器的大小，看看是否会发生变化；另外由于设置了 alt 属性，如果将鼠标放在下面的图片上，会弹出一个黄色的小文本框，其中显示 “大卡通”的说明。



图 3-41　在网页中添加图片

3. 设置图片布局

所谓布局，就是图片放在网页中的位置，以及图片与文本的排放关系。实现这种功能，除了使用上面介绍的<IMG>标签外，还需要使用<CENTER>、<P>等标签。

(1) 设置图片的对齐方式

图片可放在页面的左边(left)、中间(center)和右边(right)。

实现图片居中的方法有两种：一是使用<CENTER>标签；二是使用<P>标签的 align 属性。格式分别为：

```
<CENTER>  <IMG  src="图片文件名">  </CENTER>
```

或者：

```
<P  align=center>  <IMG  src="图片文件名">  </P>
```

如果希望设置为居左或居右，可以使用<P>标签的 align 属性来实现，其格式为：

```
<P  align=left 或 right>  <IMG  src="图片文件名">  </P>
```

【实例 3-32】图片的不同对齐方式

下面的例子在网页中添加了同一幅图片，但对它设置了不同的对齐方式，程序代码如 ex3_32.html 所示。

ex3_32.html

```
<HTML>
  <HEAD> <TITLE>图片的不同对齐方式</TITLE> </HEAD>
  <BODY>
    <FONT size=2>
      <P align=left><IMG src="cat.jpg" alt="左边的猫" width=60 height=80>居左</P>
      <P align=CENTER><IMG src="cat.jpg" alt="中间的猫" width=60 height=80>居中</P>
      <P align=right><IMG src="cat.jpg" alt="右边的猫" width=60 height=80>居右</P>
      <CENTER><IMG src="cat.jpg" alt="中间的猫" width=60 height=80>居中</CENTER><BR>
    </FONT>
  </BODY>
</HTML>
```

运行后的浏览器显示如图 3-42 所示，图中图片出现了四次，前面三次使用了<P>标签的 align 属性来控制图形的位置，最后一个使用了<CENTER>标签。

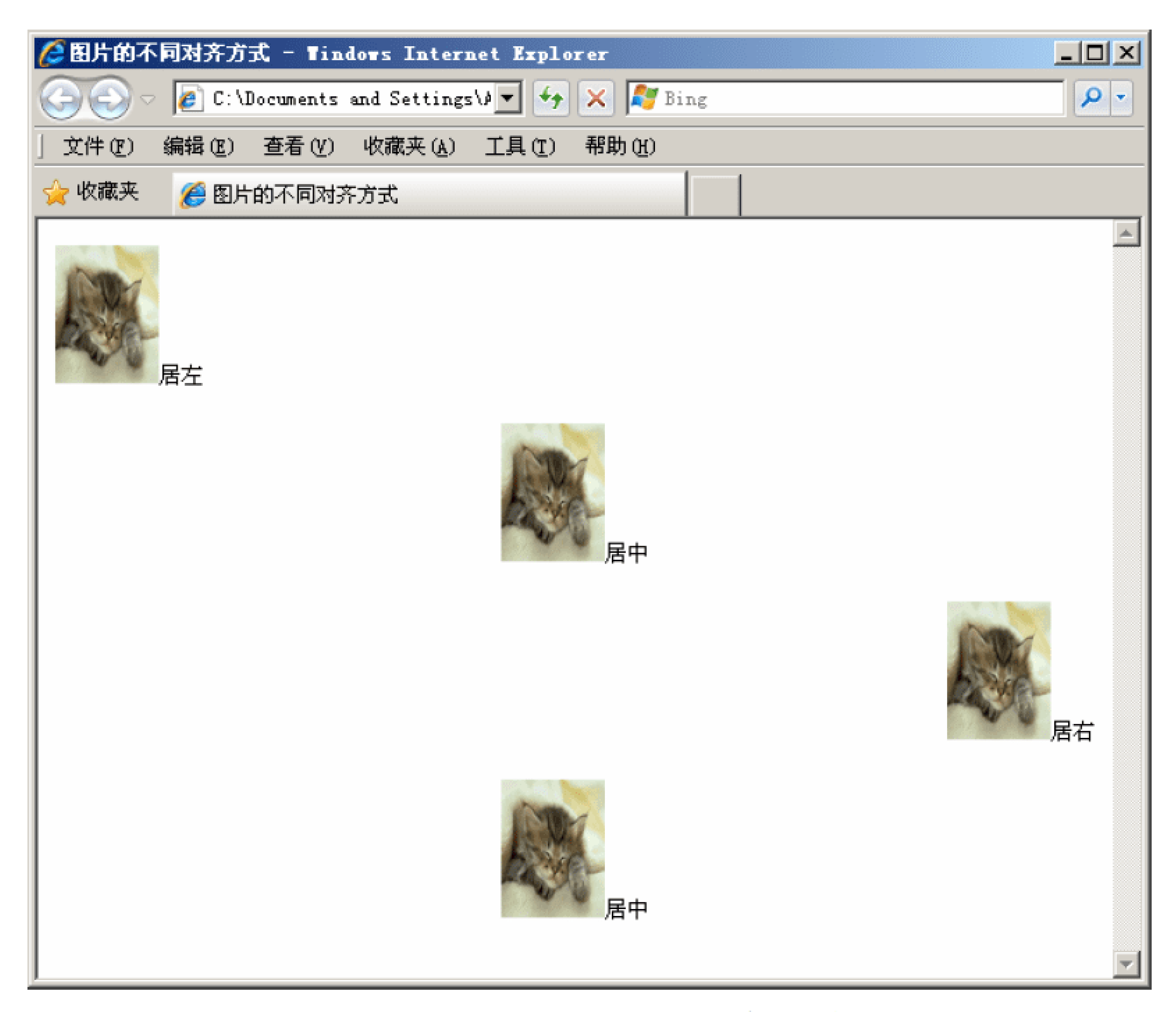


图 3-42　图片的不同对齐方式

(2) 设置图片与文本间的关系

可以设置图文的混排版面，如其间的空白、图文的对齐、文本环绕图片等。有以下几种常用方法。

① 设置图片与文本之间的空白。适当在图片与文本之间留下空白，可使页面看起来不至于太紧密。<IMG>标签的 hspace 和 vspace 属性可实现该功能。

【实例 3-33】设置图片与文本之间的空白

下面的例子对图片周围的空白进行了设置，程序代码如 ex3_33.html 所示。

ex3_33.html

```
<HTML>
  <HEAD> <TITLE>设置图片与文本之间的空白</TITLE> </HEAD>
  <BODY>
    <FONT size=4 color=blue><CENTER><B>设置图片与文本之间的空白</CENTER> </FONT>
    <IMG src="cat.jpg" width=100 height=140 hspace=5 vspace=5>猫 1
    <IMG src="cat.jpg" width=100 height=140 hspace=20 vspace=20>猫 2
    <IMG src="cat.jpg" width=100 height=140 hspace=40 vspace=40>猫 3
    <IMG src="cat.jpg" width=100 height=140 hspace=20 vspace=20>猫 4
    <IMG src="cat.jpg" width=100 height=140 hspace=5 vspace=5>猫 5
  </BODY>
</HTML>
```

运行后的浏览器显示如图 3-43 所示，图中文字的高度是相同的，而图片的位置却因为 hspace 和 vspace 的设置而有所变化，请读者仔细观察。



图 3-43　设置图片与文本之间的空白

② 文本与图片在垂直方向上的对齐。图片与文本混合排放时，利用<IMG>标签的 align 属性可设定它们在垂直方向的对齐。格式如下几种。

```
文本  <IMG  src="图片文件" align="对齐方式">  文本
```

其中 align 的值可取如下几种。

- top：文本与图片的顶部对齐；
- middle：文本与图片的中央对齐；
- bottom：文本与图片的底部对齐。

【实例 3-34】文本与图片在垂直方向上的对齐方式

下面的例子说明了文本与图片在垂直方向上的对齐方式，程序代码如 ex3_34.html 所示。

ex3_34.html

```
<HTML>
  <HEAD> <TITLE>文本与图片在垂直方向上的对齐方
```

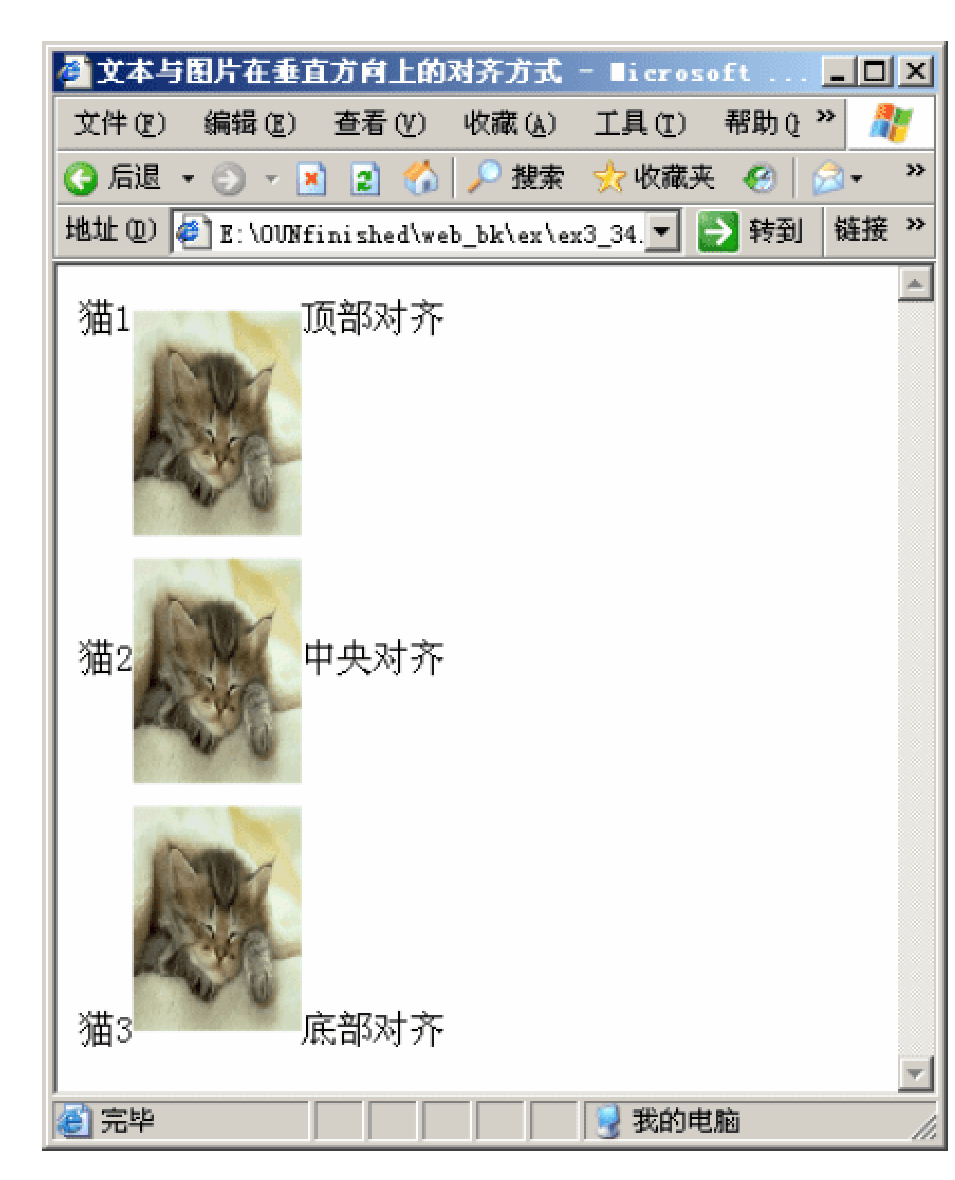


图 3-44　文本与图片在垂直方向上的对齐方式

```
式</TITLE> </HEAD>
    <BODY>
      猫 1<IMG src="cat.jpg" width=75 height=100 vspace=5 align=top>顶部对齐<BR>
      猫 2<IMG src="cat.jpg" width=75 height=100 vspace=5 align=middle>中央对齐<BR>
      猫 3<IMG src="cat.jpg"   width=75 height=100 vspace=5 align=bottom>底部对齐<BR>
      </FONT>
    </BODY>
  </HTML>
```

运行后的浏览器显示如图 3-44 所示，请读者仔细观察图中文本与图片之间的相对位置，体会文本与图片在垂直方向上的对齐方式。

③ 文本环绕图片。如果不设置文本对图片的环绕，图片在页面会占一片空白。利用<IMG>标签的属性，可以使文本环绕图片。格式为：

```
<IMG   src="图片文件名"   align="环绕方式">   文本
```

其中 align 的值可取如下几种。

- left：图片居左，文本在图片的右边；
- right：图片居右，文本在图片的左边。

使用该标签设置文本环绕方式后，将一直有效，直到遇到下一个设置标签。如果想取消文本环绕图片，可使用<BR clear>标签，其后的文本将不再环绕图片。格式为：

```
<BR clear=left 或 right 或 all>
```

其中 clear 的值可取如下几种。

- left：解除在图片左侧放置的文本；
- right：解除在图片右侧放置的文本；
- all：解除在图片左、右侧放置的文本。

【实例 3-35】文本环绕图片的设置与解除

下面的例子说明了文本与图片环绕的设置与解除，程序代码如 ex3_35.html 所示。

ex3_35.html

```
<HTML>
  <HEAD> <TITLE>文本环绕图片的设置与解除</TITLE> </HEAD>
  <BODY bgcolor=ivory>
    <H4 align=center>加菲猫</H4>
    <FONT size=3 color=blue>
      <IMG src="cat.jpg" width=100 height=140 align=left>
      我们热爱的加菲猫已经 26 岁啦！但它还来不及为青春逝去而黯然神伤，<br>
      而是兴致勃勃地随主人乔恩(布瑞金 • 梅耶饰)来到了英国旅行。<br>
      <IMG src="cat.jpg" width=100 height=140 border=3 align=right>
           
      在伦敦，一次偶然的事件中，加菲猫被误认为是一只长得十分相似的皇室
      血统猫，并因此意外继承了一座美丽的城堡。在这座城堡里，加菲猫终于过上了
      它梦寐以求的生活——享受管家和仆人的照顾，和朋友们一起狂欢 party，每
      天可以睡上好多个午觉，醒来后还有可口的下午茶搭配无限量供应的猪肉卷……
```

```
          <BR clear=all>
          <IMG src="cat.jpg" width=100 height=60   hspace=10 vspace=5>
              
          加菲猫，品种属橘色虎斑猫，1978 年 6 月 19 日生于一家意大利餐馆的厨房…<BR>
        </FONT>
      </BODY>
    </HTML>
```

运行后的浏览器显示如图 3-45 所示，请读者仔细观察图中文本与图片环绕的设置与解除所产生的效果，体会其中的区别。



图 3-45　文本环绕图片的设置与解除

(3) 用图片作为超链接

图片也可作为热点，单击图片则跳转到被链接的文本或其他文件。格式为：

```
    <A   href="链接到的文件名">  <IMG src="图片文件名">   </A>
```

【实例 3-36】用图片作为超链接

下面的例子说明了用图片作为超链接的方法，程序代码如 ex3_36.html 所示。

ex3_36.html

```
    <HTML>
      <HEAD><TITLE>用图片作为链接</TITLE></HEAD>
      <BODY bgcolor=#CCCCCC>
        <H4 align=center>《加菲猫》内容简介</H4>
        <FONT size=3>
          <A href="JFM.html"> <!-- 单击图片则打开 JFM.html -->
            <IMG src="cat.jpg" alt="加菲猫" width=140 height=190 align=left> </A>
                从上次“英勇”解救老被自己欺负的小狗欧弟，加菲猫已经很久
没有劳筋动骨了！睡觉、吃饭、躺在椅子上发困，和主人乔恩抢遥控器，再小小地鄙视一下头脑简单
的欧弟，懒惰的“猫生”总是很美好！不过，主人乔(布瑞金·梅耶)为了，心爱的莉丝(珍妮弗·拉芙·海
威特)决定飞到英…… </FONT>
      </BODY>
    </HTML>
```

运行后的浏览器显示如图 3-46 所示，请读者仔细观察，当鼠标位于图片的位置时，鼠标变为手的形状，同时出现提示“加菲猫”，浏览器左下角的状态栏中出现了链接的地址，单击后会转向这个地址。

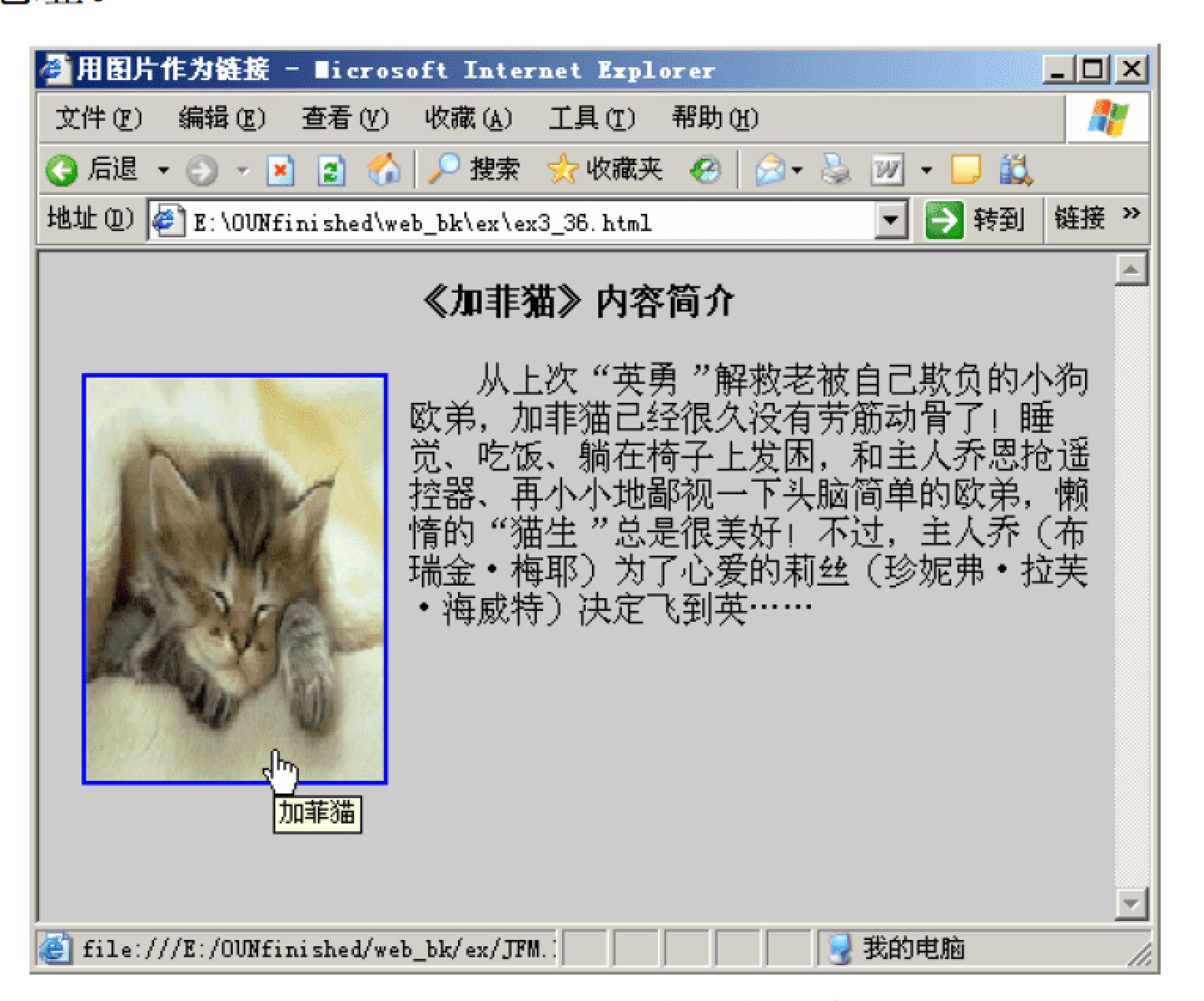


图 3-46 用图片作为超链接

(4) 在图像上定义热区

使用 Image Map 这项技术可实现在一幅图像上定义多个热区，每个热区指向一个相应的 URL 地址。当使用者单击不同的区域，就可以链接到不同的地方。对此，需要在<IMG>标签上定义 usermap 属性，然后在<MAP>标签中再嵌套定义标签<AREA>…</AREA>来实现。

其中在标签<AREA>上，需要使用的属性如下。

- Shape：定义形状，属性值可为 rect、circle、polygon、default。
- Coords：定义一个以逗号分隔的坐标列表。

热区的使用方法如下：

```
<IMG SRC="KeyGif.gif" usemap="#myMAP">
<MAP name="myMAP">
<AREA shape="circle" coords="30,30,30" href="temp.html">
<AREA shape="rect" coords="10,10,100,100" href="temp.html">
</MAP>
```

上面的语句定义了一个名为“myMAP”的地图，其中包含了两个热区，一个为圆形，一个为矩形，单击后链接到“temp.html”。

【实例 3-37】在图像上定义热区

下面的例子说明了在图像上定义热区的方法，程序代码如 ex3_37.html 所示。

ex3_37.html

```
<HTML>
  <HEAD><TITLE>在图像上定义热区</TITLE></HEAD>
```

```
    <BODY bgcolor=#CCCCCC>
      <IMG SRC="rd.jpg" USEMAP="#mymap">
      <MAP NAME="mymap">
        <AREA shape="circle" coords="75,75,75" alt="圆形热点" href="ex3_37.html" target="_blank">
        <AREA shape="rect" coords="10,160,150,280" alt="矩形热点" href="ex3_37.html"
target="_blank">
        <AREA shape="poly"
coords="180,130,300,100,360,140,370,200,360,270,
180,250" alt="多边形热点" href="ex3_37.html"
target="_blank">
      </MAP>
    </BODY>
  </HTML>
```

运行后的浏览器显示如图 3-47 所示，请读者仔细观察，图片上标注了三个热区的大致位置，当鼠标位于图片的不同热区时，鼠标会变为手的形状，同时出现相应热区的提示，并在浏览器左下角的状态栏中显示链接的地址，单击后会转向这个地址。



图 3-47　在图像上定义热区

3.4　本章小结

本章讲解了一个重要的部分——HTTP 和 HTML 的知识，对于 HTTP 本书分析了它的基本原理，HTML 的介绍是本章的重点。对此本章系统地介绍了标题和段落、文字、列表、超级链接、表格、图像等方面常用的元素及其属性的基本用法。

3.5　思考和练习

1. 能否通过手工在键盘上输入上述的 HTTP 请求来获取 Web 服务器的响应？

2. 与 HTML 相类似的语言标准有不少，比 HTML 功能强大的语言标准也早已存在，为什么 HTML 能脱颖而出，成为一种在 Web 上流行的语言？

3. 如果 HTML 格式的文件不保存成扩展名为 html 的文件，是否可以在浏览器中看到正确的结果？

4. 如果对<BODY>的属性设置的顺序颠倒了，结果是否会出现差异？

5. 如果希望表格中的表项能随窗口的变化而自动变化，该如何设置？

第4章　交互设计与HTML高级应用

现在的网站已经提供了越来越多的交互操作，因此对于网站的交互设计是一个不得不重视的问题，本章首先介绍了网站的交互设计方法，通过原理的介绍和实例的剖析明确了交互设计的基本原则和方法。此外，HTML 中还包括如框架、表单、脚本、各种动态效果和多媒体等方面的高级应用，它们可以使网页以更丰富的形式展示在用户面前。本章中介绍了针对这些高级应用所涉及的元素和属性的用法。另外，本章还介绍了最新的 HTML5 的有关用法。

本章要点：

- 交互设计
- HTML 高级应用
- HTML5 介绍

4.1　网站的交互设计

交互设计，又称互动设计(英文 Interaction Design, 缩写 IxD 或者 IaD)，是定义、设计人造系统行为的设计领域。人造物，即人工制成物品，例如，软件、移动设备、人造环境、服务、可佩带装置以及系统的组织结构。

交互设计在于定义人造物的行为方式(the “interaction”，即人工制品在特定场景下的反应方式)相关的界面。交互设计作为一门关注交互体验的新学科，在 20 世纪 80 年代就产生了，当时 IDEO 的创始人——比尔·摩格理吉(Bill Moggridge)在 1984 年一次设计会议上提出，他一开始给它命名为“软面”(Soft Face)，由于这个名字容易让人想起当时流行的玩具“椰菜娃娃”(Cabbage Patch doll)，所以他后来又把它更名为“Interaction Design”，即交互设计。

网站是众多人造系统中的一种，在网站与用户之间的交互越来越复杂的今天，网站的交互设计也愈发变得不可或缺。

4.1.1　交互设计

需要进行交互设计的前提来自于：一个网站本身是没有生命力的，它需要被赋予对于各种行为的反馈机制；将用户所期望的反馈赋予给它，让它给出合理的反馈行为。简单来说，网站的交互设计就是以满足用户的需求为目标，通过对用户期望和需求的理解，在一定的技术和业务框架下，所创造的形式、内容、行为、反馈方式等，它们能满足有用和易

用性。

随着网络的成熟和发展，以追求技术的新颖性和技巧性的网站设计思想被演变为以用户为中心的设计思想，网站的可用性和易用性逐渐成为网站能否吸引访客的要点。实际上，设计一个网站并不难，但是要设计一个让目标客户能够乐在其中，并且能从网站上很容易找到自己希望寻找的内容的网站则是一门学问。网站如何去提供一个好的交互界面，并让网站的目标客户能够很好地认同网站的价值，才是网站设计的最终目标。

显然，交互设计不仅仅是美工设计，良好的视觉效果并不是网站交互设计的全部。网站的交互设计大致有以下几个层次，如图 4-1 所示。

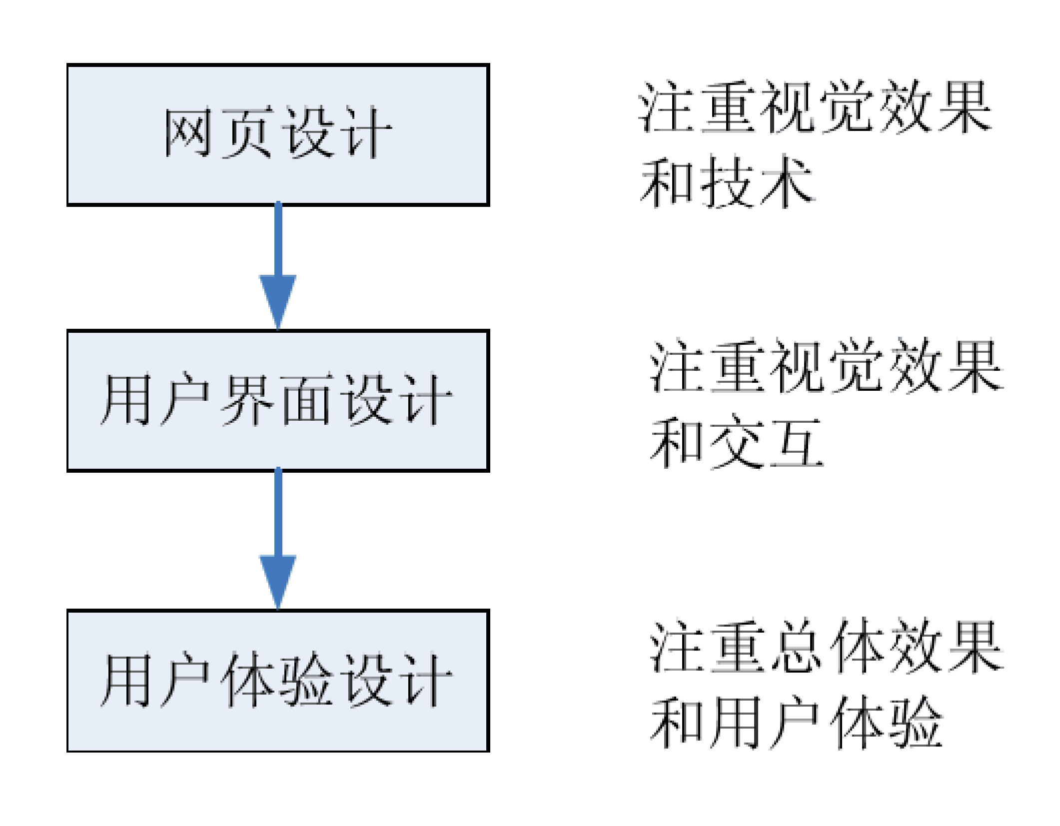


图 4-1　交互设计的不同层次

在设计时需要首先考虑目前的技术和方法能达到什么层次，最终的要求是什么，两者之间是否存在差距等。交互设计的工作实际上从战略层的布局开始，一直到后期围绕结构、框架以及表现层的所有工作，是贯穿全局的。为了达到这个目的，通常有两种理念，即基于用户行为体验的和基于用户情感体验的两种。

1. 基于用户行为体验的交互设计

用户的行为体验是建立在用户的交互需求的基础上的，交互需求是人与产品或者系统交互过程中的需求，包括完成任务的时间、效率，是否顺利，是否出错，是否有帮助等。可用性研究关注的是用户的交互需求包括一件产品在操作时的学习性、效率性、记忆性、容错率和满意度等。交互需求关注的是交互过程是否顺畅、用户是否可以简单地完成他们的任务。

Jokela 在 Kano 的质量模型基础上，提出产品的三种可用性：必须有的、更多即更好的以及具有吸引力的，这三种可能性都会影响用户的满意度。

- “必须有的”可用性代表用户期望从一件产品中获得的可用性，也就是产品应该具有的最基本的可用性。如果该产品中没有出现“必须有”的要素就会导致用户不满意度的出现。
- “更多即更好”可用性对用户满意度具有线性影响，即这种可用性越高，用户就越满意。
- “具有吸引力”可用性测试会让一件产品从其他产品中脱颖而出，提供较高的用户满意度，如 iPhone 的交互模式。

一个网站要想获得成功，至少要获得“必须有的”可用性。这意味着可用性虽然不是竞争力的决定因素，但却是提高用户满意度的必要条件。“更多即更好”可用性则意味着网站在新的特征方面能与竞争保持一致。“具有吸引力”可用性是为了实现从竞争对手中脱颖而出的目的。现在很多新产品，都很注重走市场差异化路线，而这种差异化是“具有吸引力”可用性所产生的结果。

有人认为，好的交互界面应当是这样的，用户能集中精力完成任务而不被界面中无关紧要的设计所打扰，即让用户感觉不到交互的存在，感受的只是流程交互的通畅、自然。多通道用户交互界面并不需要显式地说明每个交互成分，反之是在自然的交互过程中隐含地说明。隐喻给人以可预测性，用户能够轻易地理解设计的软件应用。它可以带给用户一种掌握及控制一切的感觉。当用户操作时，他们知道下一步即将出现什么、怎么回去——即使是在第一次操作，具体的实例如图 4-2 所示。

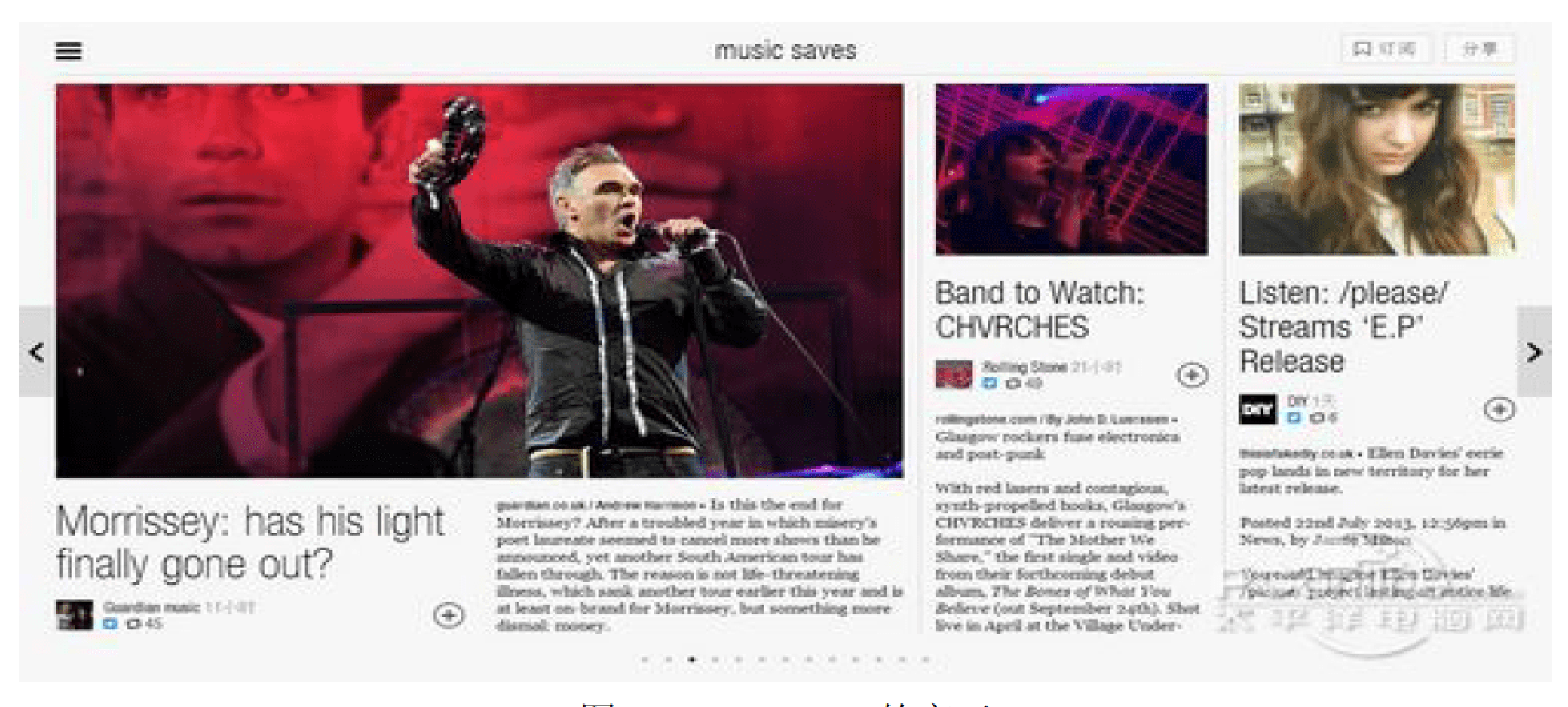


图 4-2　Filpboard 的交互

用户只需用鼠标或触摸板点击网页即可左右翻页，点击底部的导航栏可以实现快速浏览翻页。

综上所述，对网站用户交互的设计，既要满足隐喻性，又要满足技术性，这样的交互才是以用户为中心的交互，也是设计者所追求的。

2. 基于用户情感体验的交互设计

用户的情感体验是建立在用户的情感需求基础上的。情感需求是人在操作产品或者系统过程中所产生的情感。如从网站本身或使用过程中感受到的关爱、互动和乐趣。情感强调的是设计感、故事感、交互感、娱乐感和意义感，对于网站而言就是需要有吸引力和趣味性。

在技术发展不成熟的年代，功能是交互设计的核心，而当技术日趋成熟的今天，人性化已经发展为设计的主题。随着技术的不断加速发展，新技术已经从各方面以更快的方式渗透到人们的日常生活中，人们对产品或者交互系统有了更多的要求。用户在考虑其感官体验、行为体验之外，更加注重使用过程中所产生的情感共鸣，更加关心系统是否具备其他一些品质，如令人满意、令人愉悦、有趣、引人入胜、富有启发性、可激发创造性和让人有成就感等情感上的满足。尤其是青少年消费者，时代给他们烙上了叛逆、自我、自恋

等标签，对产品的要求不仅仅是能用的功能，更重要的是新奇、刺激、个性、潮流。反映在设计上就需要个人的表现风格和审美。所要求交互越来越丰富，能体现出人情味和与众不同，而非冷冰冰的功能组合。不仅仅要求新奇的功能，还要求满足人的审美和认知。

交互的布局、色彩、互动过程等，都需要充分考虑人的需求，让人没有接触一个冰冷、生硬的交互的感觉，而是在使用一个似曾相识的现实生活中的物件，让人有亲切感，这就是人性化的表现。把感情通过交互传递给人，取得与人的共鸣。这种感情的传达是确定性与不确定性的统一。在设计的过程中，设计师很容易因为太富个性而加入自己的情感因素，而忽略了用户的情感需要，这样很容易设计成一幅艺术创作，违背了产品设计的初衷。主观操作复杂性越低，即系统越容易被使用，则说明系统的用户友好性越好。

趣味性和幽默感也是人的本性之一，而具有趣味性和幽默感的事物同样能够直接激发人的本能情感。交互设计中所讲的“趣味性和幽默感”是指图标、图形、字体等的形态、色彩、形式等方面看起来很有趣、有意思、有味道、能引人注意，或者呈现出圆润、憨厚、笨拙可爱、亲和力等特别使交互具备趣味性和幽默感的特征，它追求的是不同寻常或出乎意料的情景，激发人们的好奇心，并能在转眼间带给人一种惊喜、快乐的心理感受，这也属本能情感。趣味性可以简单地理解为某事物的内容能使人感到愉快，并且能引起人们产生兴趣的特性，从而达到增强用户注意力、兴趣度和记忆力的目的，使得产品更有新意、更有吸引力和亲和力。因此，对网站的交互设计，既要满足趣味性，又要满足幽默感。

4.1.2　设计原则和方法

几年前，IBM 发现其网站运作得并不好，经初步分析表明，搜索功能用得最多，而帮助功能却成为第二有用的功能。但 IBM 的网站却让用户很难找到搜索功能；且由于搜索功能找不着，很多访问者使用帮助页面寻求帮助。此后 IBM 重新设计了网站，使其导航更为清晰。在重新设计的网站上线一周以后，依赖于帮助页面和搜索功能的用户数量明显下降，而在线购物量提高了400%，这个实例从一个侧面说明了交互设计的重要作用。以下给出了网站交互设计的基本原则：

- 如果可能，网站可以为每个用户提供个性化的内容。
- 提供“为什么要成为网站用户”的提示并提供一定的奖励。
- 让用户尽可能多地接触到网站推送的产品或内容。
- 不要让用户感到有些产品/服务正在强迫他们购买或使用。
- 在适当的时候可轻松地访问重要的部分或内容。
- 在任何时候让顾客有良好的使用体验并对使用过程可控。
- 电子商务等类型网站应将焦点放在产品搜索和在线购买上。

比如苹果网站的网上商城，在订购的时候，看起来是和大多数网站一样的搜索框，但是输入查找的内容时，搜索框的右侧会出现一个橘黄色的按钮，点击之后搜索框里刚刚输入的内容被清空了，可是大多数其他类似网站却需要反复点击“Backspace”键，这些设计细节往往成为提高用户体验度的亮点。

交互设计是一个过程，它不仅仅是画线框图。交互设计最关键的两个环节是页面流程

和页面布局，前者建立清晰的架构和严密的逻辑，后者整合零散的信息并确定分明的主次关系。这一切都是为了我们的终极目标——让界面符合用户的预期，不带给他们任何的意外。一切都在用户的意料之中。

一提到交互设计，就会使人联想到画线框图或原型图。实际上，交互设计是一个过程，从开始到结束有一套系统的流程，而原型图只是其中的一个环节。

交互设计流程中主要的步骤包括：任务分析(列出界面所要完成的所有任务)、按各任务确定页面流程(建立信息架构)、创建统一的页面布局、在页面布局的基础上进行原型设计、编写设计说明等。

1. 任务分析

第一步任务分析。这里要做的是对于将要设计的这个新界面的所有任务的分析，也就是用户在界面上能进行的所有操作。这个分析在功能需求的基础上进行，需求方一般提供一个功能点的列表。

任务分析最常见的是任务列表，另外有任务流程和任务场景等。下面以任务列表为例。

列出所有主要任务，以及每个主要任务的子任务。再把子任务细分到各个步骤。形成下面这个列表。

```
主要任务 1
    子任务 1
        步骤 1
        步骤 2
    子任务 2
        步骤 1
        步骤 2
主要任务 2  …
```

以个人相册为例，经任务分析后形成的列表如下：

```
相册列表
浏览相册
    浏览相册列表
    选择相册
    浏览照片
创建新相册
    添加照片
        选择已有相册
        选择照片
        排列顺序
        添加字幕文字
        选择动画效果
添加模板
    浏览模板
    选择模板
添加音乐
    浏览音乐列表
        试听音乐
```

```
                选择音乐
            增加新音乐
                打开本地文件
                选择音乐
        相册预览(略)…
        相册 命名(略)…
        保存相册(略)…
        修改相册(略)…
```

任务列表包括所有功能点，并对每一个功能点的逻辑关系进行整合，有时会对各任务的使用频率和其他影响设计的重要因素做进一步的分析。

2. 页面流程设计

任务分析完成后，进入设计的第一步，即设计页面流程。页面流程是设计的开始，也是重要的一环。它决定整个界面的信息架构和操作逻辑。

页面流程是上一步任务分析的自然转化。一般来说，一个主要任务就是一个页面，其他子任务也可以转化为页面。

以个人相册为例，页面流程示意图如图 4-3 所示。

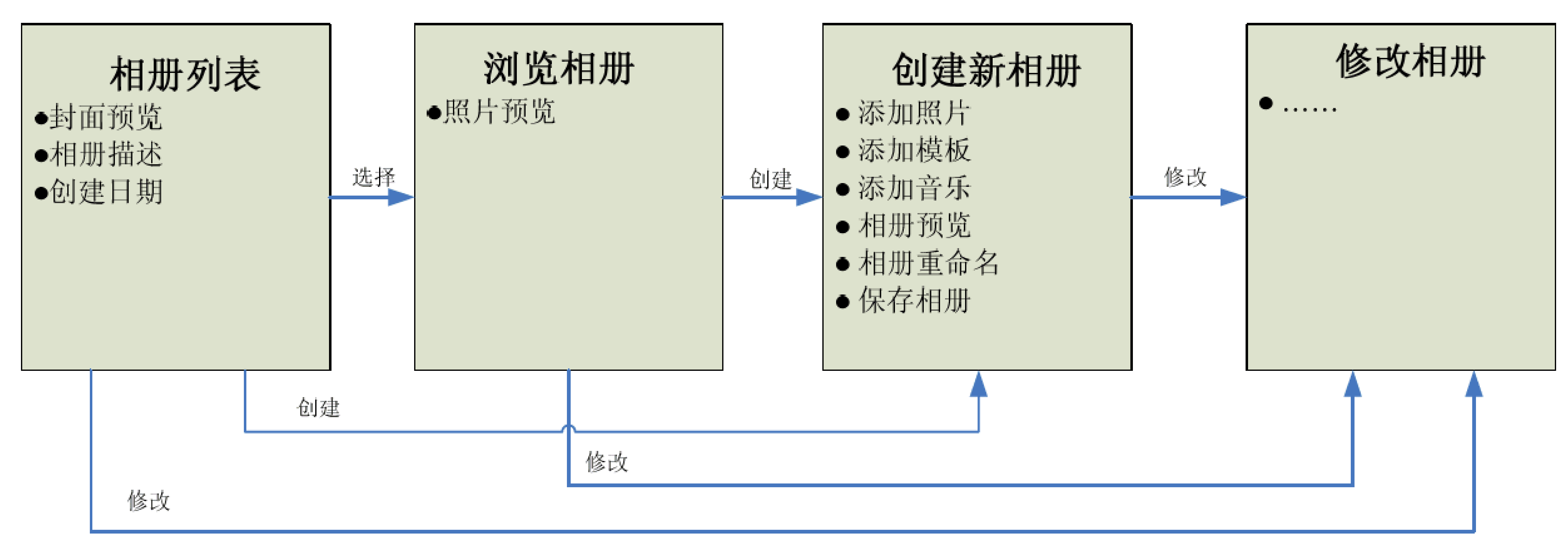


图 4-3　个人相册的设计流程示意图

页面流程表达了任务分析的结果，注意这个流程图应包括所有的页面，通过这个图可以看清页面的数量、页面的主要内容以及各页面间的关系。

3. 页面布局设计

这是具体页面设计的开始，在第 2 步知道有哪些页面需要进行设计后，这里对页面进行划分，对内容进行组织，其中最重要的一点是确定页面分区。

以个人相册为例，首先需要确定总布局，即通用布局，适合所有页面，个人相册的总布局如图 4-4 所示。

具体页面布局，在不与总布局冲突的情况下，有更细节的布局，如图 4-5 所示。

页面布局赋予零碎的内容以逻辑性，以分区的形式把页面各区域所对应的功能区确定下来，减少具体设计时的随意性。这是设计严谨与否的表现所在。把类似的操作放在一起，对于用户来说是可以预见的，用户能够判断哪个操作在哪个区域，减少盲目寻找带来的困难和疑惑。

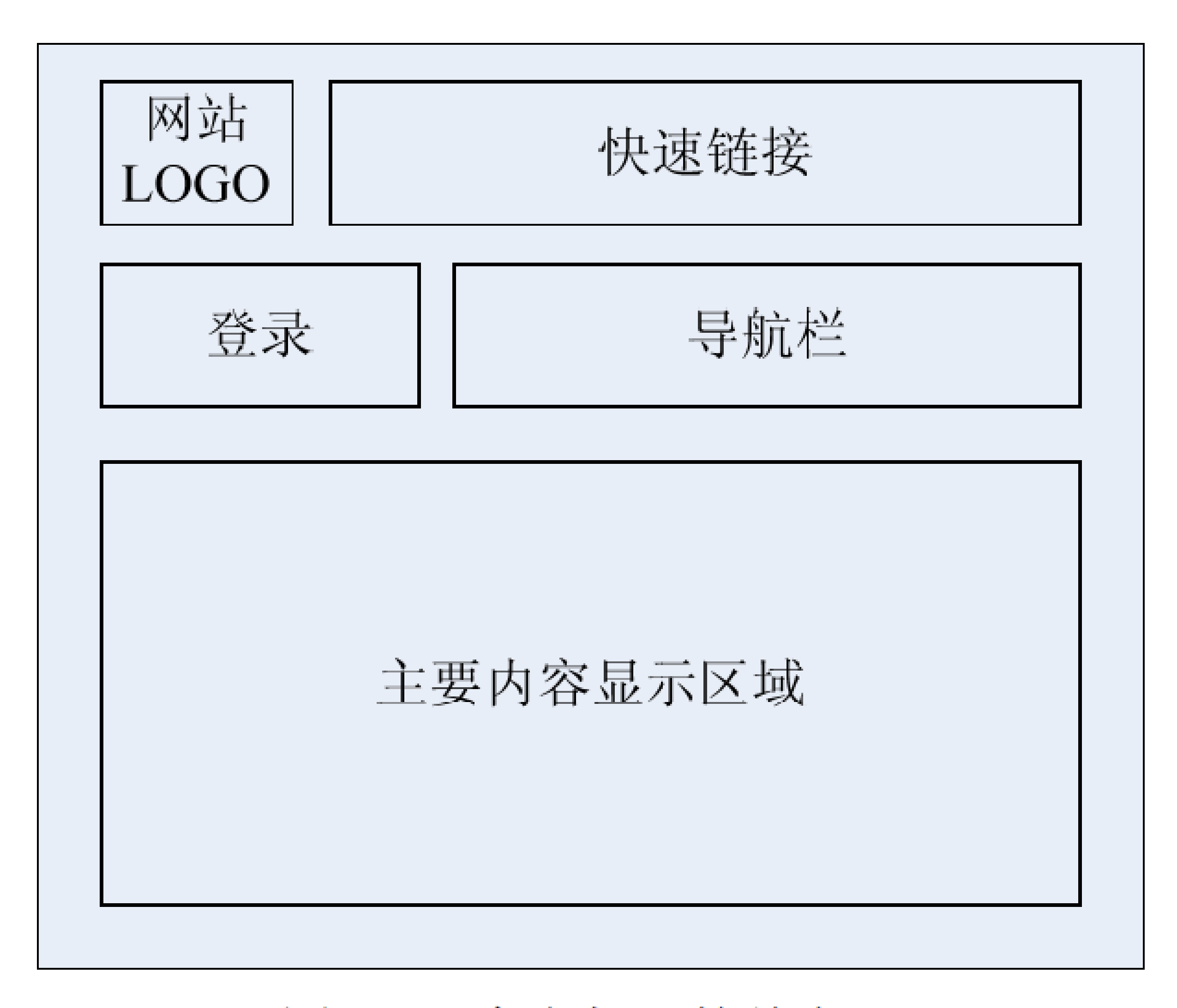



图 4-4　个人相册的总布局

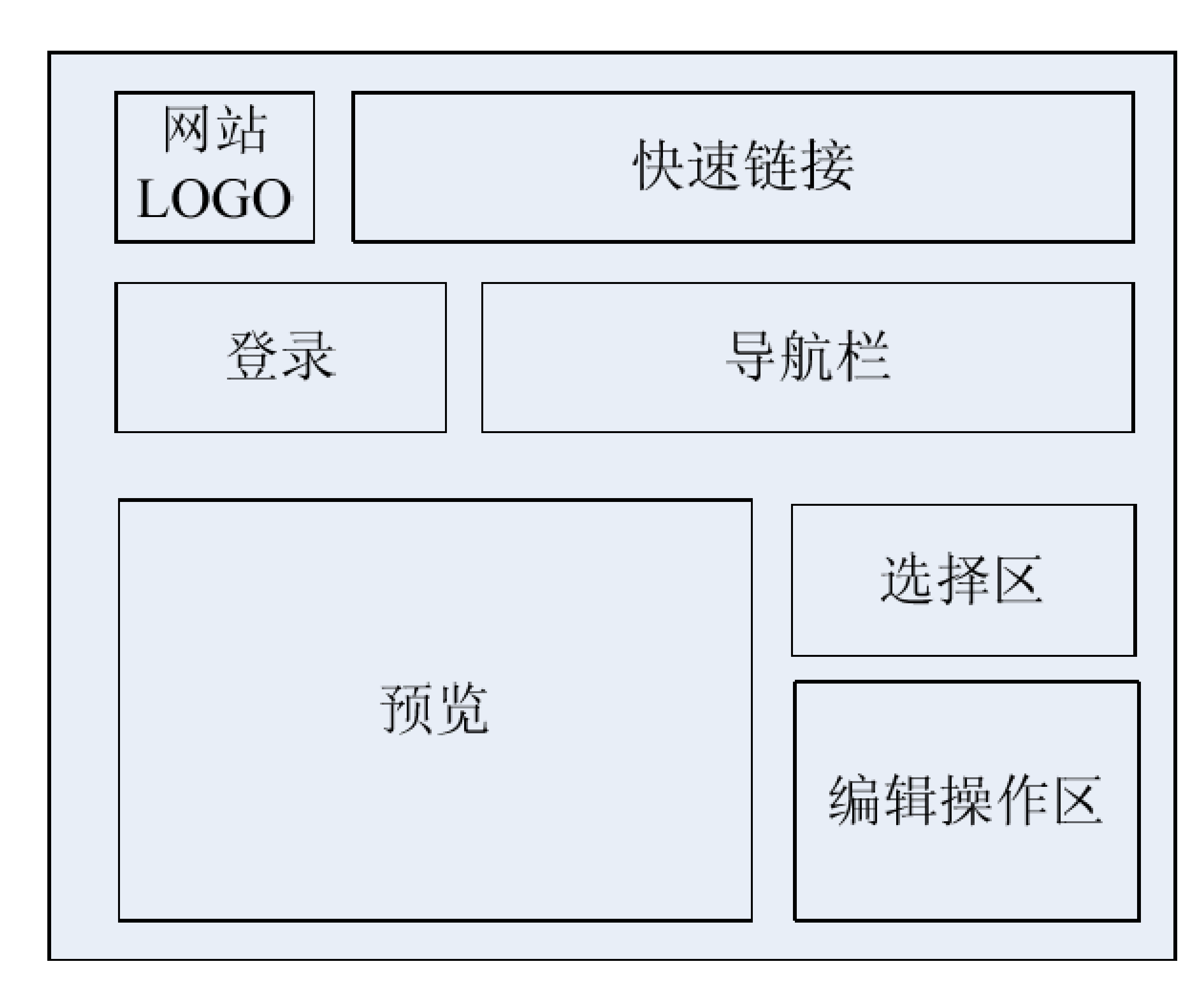


图 4-5　个人相册的具体页面布局

4. 原型设计

这一步是大家熟知的，即具体页面的设计。这一步设计把所有的界面元素表现出来，可以有低保真和高保真原型图。低保真即线框图，高保真多是接近最终效果图。个人相册的原型图如图 4-6 所示。



图 4-6　个人相册的原型图

5. 设计说明

最后一步需要做的是对所有页面进行详细的描述，包括对页面上所有元素进行说明，比如默认状态、跳转页面、字号字体、尺寸等。这是将要交给开发人员的文档，以及测试人员后期进行测试的依据。

4.1.3　一个交互设计的实例

实现同一个目的，有多种不同的方案，每个方案，都可能在这个方面强一些，在另一个方面弱一些；实际使用上，它们的效果通常并没有非常明显的区别。以下是一个用户购票系统交互设计的实例。

交互设计在实现细节方面要达到既要易用流畅，又要屏蔽干扰项，还能方便用户随时切换，且充实又简洁等。当然，完全符合这些特征的设计方案可能是不存在的，但为了网

站的总体目标，需要尽可能多地满足这些特征。这时区分不同方案之间的差异，了解各自间有什么优劣之处，往往可以帮助决策者快速地在不同的方案间进行取舍。下面以一个机票查询的例子来进行说明。

大多数机票查询的交互方式是类似于图 4-7 所示。其中单程/往返使用两个 radio，放在最前面；选择单程时禁用往返，或者隐藏返程时间输入框。

其设计逻辑是这样的：用户最先决定自己是单程还是往返，然后再选择起降地点及时间……这显然是产品逻辑，实际上绝大多数用户都不是一去不回，而是必须回来的。用户随时可能从买单程票变成想买往返票，而这时候再把注意力返回到页面最上方，又因为离得很远而成为负担。而不禁用返程，又担心用户被这个框干扰，影响任务的继续。

于是一个改进的版本如图 4-8 所示。其中返程时间不禁用，只是视觉相对弱化；返程时间输入框获得焦点时，上方 radio 自动切换到往返；如果要回到单程状态，仍然只能手动通过 radio 切换。



图 4-7　某个机票查询交互实例



图 4-8　机票查询交互顺序的重新整合

这个控件组的作用，抽象的说，就是二选一的入口，分别对应不同的内容展示：两个 radio/下拉列表/tab，都能起到二选一的作用。下拉列表鼠标操作次数太多，适用于更多选项的情形；tab 带有两者平行且无交集的意味，所以在这种情况下也不是很适用。

对此问题进行进一步的思考，发现这是一种特殊的二选一问题，也就是是否需要返程，于是设计方案可以改进为采用复选框，设计结果如图 4-9 所示。

这时，用户在往返与否之间切换的成本最低。当然这并不是说最开始使用 radio 的方案存在错误，只是在只有这两种情况时，使用复选框能更好地实现设计的总目标。

当然，如果存在更多的选项，此时采用 radio 会更好一些，如图 4-10 所示。

总之，交互设计的目的就是“以人为本”，一个好的交互设计不仅能让网站变得有个性、有品味，还能让操作过程变得舒适、简单、自由，充分体现软件的定位和特点。只要达到了这些目标的交互设计都是好的设计。



图 4-9　采用复选框的机票查询交互实例



图 4-10　在更复杂的交互中采用 radio 的方案

4.2　HTML 高级应用

4.2.1　框架

框架(FRAMES)是一种在同一浏览器窗口中显示多个相互隔离的 HTML 页的结构，它首先分割显示区域，再在每个区域中显示某个指定的文件。利用框架结构可以实现在一个页面中同时浏览多个页面，还可以在一个区域中显示所有页面的总索引，通过单击该区域中的超链接，将相关网页显示在另一个区域中。使用框架非常直观，使用户在浏览局部内容时，仍对整个网站的结构有清晰的认识，不至于进入多层链接后而迷失方向。

注意：

在 HTML 5 中不支持<FRAME>框架了，因此使用时需要注意。如果需要使用，可以采用 DIV+CSS，再配合 iFrame 的实现方式。

此外，还可以利用在本书第 5 章中将要介绍的 CSS 技术来实现框架，当然实际应用过程中需要根据网站的目标和技术框架来选择合适的方案。

1. 建立框架

框架的建立使用<FRAMESET>、<FRAME>两个标签。所有框架标签需要放在一个总起的 HTML 文件中，这个文件只记录了该框架如何划分，不会显示任何资料，所以只是以<FRAMESET>标签取代<BODY>标签，用来声明框架的定义。<FRAMESET> 用以划分框窗，每一框窗由一个<FRAME>标签所标示，<FRAME>必须在<FRAMESET>范围中使用。<FRAME>标签用来声明其中框架页面的内容。建立框架的基本结构为：

```
<FRAMESET>
```

```
<FRAME   src="URL">
<FRAME   src="URL">
…
</FRAMESET>
```

以下对上面的定义进行详细的说明。

(1) <FRAMESET>标签

<FRAMESET>标签用来定义一个框架组的属性，其格式为：

```
<FRAMESET  row="数"  cols="数"  border="像素数"  bordercolor="颜色"  frameborder="yes/no"
framespacing="值"> … </FRAMESET>
```

其中属性说明如表 4-1 所示。

表 4-1　<FRAMESET>标签中使用的属性及其说明

属 性 名 称	说　明
row	设定横向分割的框架数目
cols	设定纵向分割的框架数目
border	设定边框的宽度
bordercolor	设定边框的颜色
frameborder	设定有/无边框
framespacing	设置各窗口间的空白

框架有横向和纵向之分。对一个框架来说，其大小是很重要的。<FRAMESET>的 rows 和 cols 属性用于设定横向分割和纵向分割的框架数目。例如，要建立有三个横向框架的页面，可写为：

```
<FRAMESET   row=x1 或 y1%或 z1*, x2 或 y2%或 z2*, x3 或 y3%或 z3*>
```

其中对于 rows 和 cols 属性的设置以及相关的说明见表 4-2 所示。

表 4-2　rows 和 cols 属性的设置

设 置 形 式	说　明
x	表示框架的绝对大小，单位是像素数，如 row=50,80,60
y%	表示框架相对于浏览器窗口大小的百分比数，如 row=80,50%,10%，这表示第 2 个横向框架的大小为浏览器当前窗口的 50%
z*	*表示自动分配。如 row=50,20%,*表示第 3 个横向框架的大小为分配了前两个框架后留下的那部分；几个*表示将窗口等分为几份，若在*前加数字，表示等分数字之几；row=2*,1*,3*表示第 1、2、3 个框架分别为整个浏览器窗口的 2/6、1/6、3/6

(2) <FRAME>标签

<FRAME>标签用于给各个框架指定页面的内容，也就是，它将各个框架和包含其内容的那个文件联系在一起。<FRAME>是一个单标签，格式为：

```
<FRAME src="源文件名.html" name="框架名" border="像素数" bordercolor="颜色"
frameborder=yes 或 no marginwidth=x marginheight=y scrolling=yes 或 no 或 auto noresize>
```

其中各个属性及其说明如表 4-3 所示。

表 4-3　<FRAME>标签各个属性及其说明

属 性 名 称	说　明
src	表示该框架对应的源文件
name	指定框架名。框架名由字母开头，用下划线开头的名字无效。例如： <FRAME src="menu.html" name="menu"> <FRAME src="../web/default.html" name="web"> <FRAME src="http://www.njupt.edu.cn/index.html" nama="n"> 也就是说，源文件可以是某个网址上的文件，只需写上 URL
border	设定边框的宽度
bordercolor	设定边框的颜色
Frameborder	设定有(yes) / 无(no)边框
marginwidth	设置框架内容与左右边框的空白
marginheight	设置框架内容与上下边框的空白
scrolling	设置是(yes) / 否 no / 自动(auto)加入滚动条
noresize	不允许各窗口改变大小，默认设置是允许各窗口改变大小的

<FRAME>标签的个数应等于在<FRAMESET>标签中所定义的框架数，并以在文件中出现的次序按先行后列对框架进行初始化。如果<FRAME>标签数目少于<FRAMESET>中定义的框架数量，则多余的框架为空。

注意：

<FRAMESET>不应当在<BODY>体中出现，否则会导致无法正常显示框架，所以不能使用<BODY>。

【实例 4-1】定义框架

下面的例子说明了定义框架的方法，程序代码如 ex4_1.html 所示。

ex4_1.html

```
<HTML>
  <HEAD><TITLE>框架演示</TITLE>
  </HEAD>
  <FRAMESET cols="120,*">
    <FRAME src="http://www.sina.com.cn/" >
    <FRAMESET rows="100,*">
      <FRAME src="http://www.sohu.com">
      <FRAME src="http://www.163.com">
    </FRAMESET>
  </FRAMESET>
```

```
    <BODY>
    </BODY>
</HTML>
```

运行后的浏览器显示如图 4-11 所示，利用框架技术，在一个浏览器界面中同时出现了三个知名网站的内容。



图 4-11　定义框架

2. 设置框架属性

框架的属性包括边框、颜色、滚动条等，这些属性使框架的外观发生变化，产生不同的艺术效果。

利用<FRAMESET>和<FRAME>标签可以设置边框的外观，如大小、颜色。两个标签所不同的是，<FRAMESET>标签设置的是整个框架各个边框的属性，而<FRAME>只能设置它所控制的框架。

(1) 设定有无边框

利用<FRAMESET>和<FRAME>标签的 frameborder 属性可以设定有无边框。格式为：

```
<FRAMESET   frameborder=yes 或 no 或 1 或 0>
或   <FRAME   frameborder=yes 或 no 或 1 或 0>
```

其中取值 yes 或 1(默认)表示生成立体的边框，而 no 或 0 表示无边框。

【实例 4-2】设置框架属性

下面的例子说明了设置框架属性的方法，程序代码如 ex4_2.html 所示。

ex4_2.html

```
<HTML>
  <TITLE> 各窗口边框的设置 </TITLE>
```

```
    <FRAMESET rows=30%,*>
      <FRAME src="Xcol.html" frameborder=yes>
      <FRAMESET cols=30%,*>
        <FRAME src="Ycol.html" frameborder=no>
        <FRAME src="Zcol.html" frameborder=no>
      </FRAMESET>
    </FRAMESET>
</HTML>
```

其中 Xcol.html 文件的内容如下：

```
<HTML>
    <TITLE>X Color</TITLE>
    <BODY bgcolor=3333ff text=ffffaa>
    <H2>X</H2>
</HTML>
```

Ycol.html 和 Zcol.html 与 Xcol.html 文件类似。

运行后的浏览器显示如图 4-12 所示，请读者仔细观察在框架上边框表现的样式；其中 X 框架是具有边框的，而 Y 和 Z 框架是没有边框的。从这个例子以及前面的【实例 4-1】可以发现，默认情况下框架是有边框的。

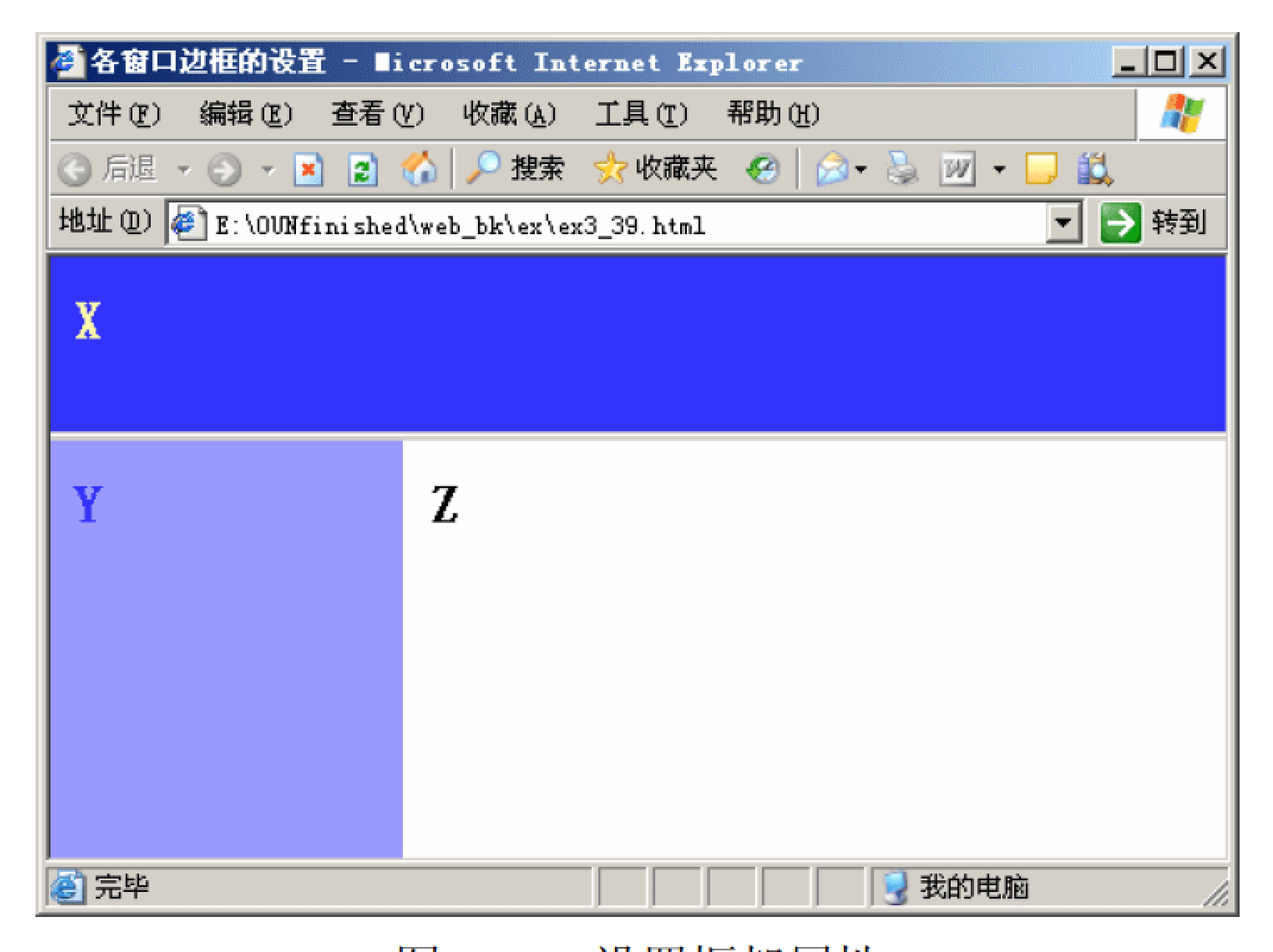


图 4-12　设置框架属性

(2) 设定边框的色彩

利用<FRAMESET>和<FRAME>标签的 bordercolor 属性给边框着色。格式为：

```
<FRAMESET   bordercolor="颜色">
或  <FRAME   bordercolor="颜色">
```

【实例 4-3】设定边框的色彩

下面的例子说明了设置框架属性的方法，程序代码如 ex4_3.html 所示。

ex4_3.html

```
<HTML>
    <TITLE> 设定边框的色彩 </TITLE>
```

```
    <FRAMESET rows=30%,*>
      <FRAME src="Xcol.html" bordercolor=red>
      <FRAMESET cols=30%,*>
        <FRAME src="Ycol.html" bordercolor=red>
        <FRAME src="Zcol.html" bordercolor=red>
      </FRAMESET>
    </FRAMESET>
  </HTML>
```

运行后的浏览器显示如图 4-13 所示，请读者仔细观察在框架上边框的红色。

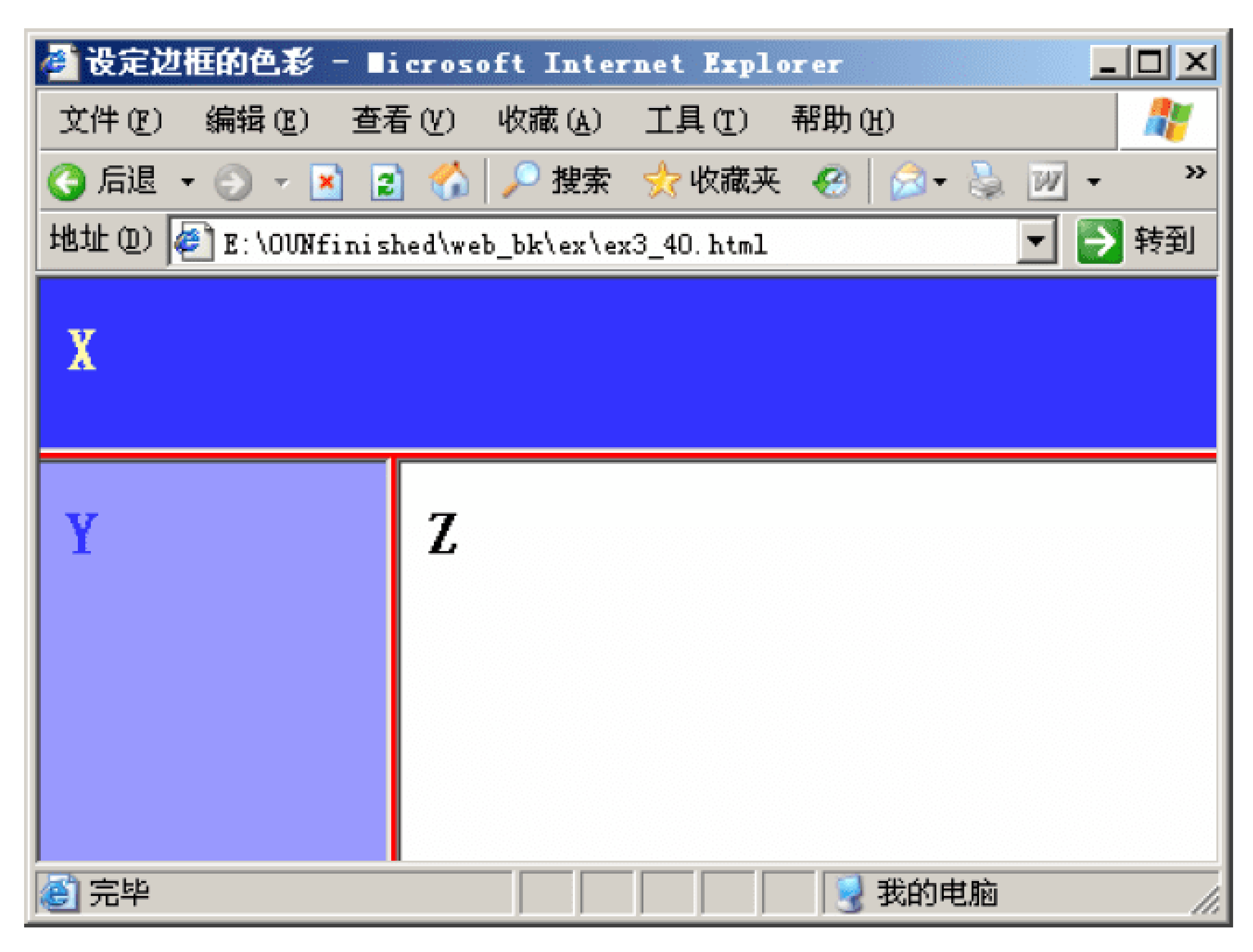


图 4-13　设定边框的色彩

(3) 固定边框

在默认情况下，用户可以用鼠标拖动边框改变页面的大小。使用<FRAME>的 noresize 属性可以固定边框的位置且不允许用鼠标拖动边框。其格式为：

```
  <FRAME  noresize>
```

其中 noresize 没有值，加入 noresize 则浏览器中边框固定，请读者自行验证。

(4) 页面空白区域的设置

框架与边框离得太近会影响美观，用标签<FRAME>的 marginwidth 和 marginheight 属性可以设框架内容与框架的左右边框间的空白。而各窗口间的空白可以用<FRAMESET>标签的 framespacing 属性来设置。

● 设置框架内容与左右边框的空白

<FRAME>标签的 marginwidth 属性可设置框架内容与左右边框之间的空白。格式为：

```
  <FRAME  marginwidth=x>
```

其中 x 为像素数，取 1 以上的值。

● 设置框架内容与上下边框的空白

<FRAME>标签的 marginheight 属性可设置框架内容与上下边框之间的空白。格式为：

```
  <FRAME  marginheight=x>
```

其中 x 为像素数，取 1 以上的值。

● 设置各窗口间的空白

<FRAMESET>中的 framespacing 属性可以设置各个窗口间的空白。格式为：

```
<FRAMESET   framespacing=x>
```

其中 x 取像素数。

【实例 4-4】页面空白区域的设置

下面的例子说明了页面空白区域的设置方法，程序代码如 ex4_4.html 所示。

ex4_4.html

```
<HTML>
  <TITLE> 页面空白区域的设置 </TITLE>
  <FRAMESET rows=30%,*>
    <FRAME src="Xcol.html" marginwidth=200 marginheight=20 noresize>
    <FRAMESET cols=30%,*>
      <FRAME src="Ycol.html">
      <FRAME src="Zcol.html" marginwidth=150 marginheight=100>
    </FRAMESET>
  </FRAMESET>
</HTML>
```

运行后的浏览器显示如图 4-14 所示，请读者仔细观察在框架中的空白区域。

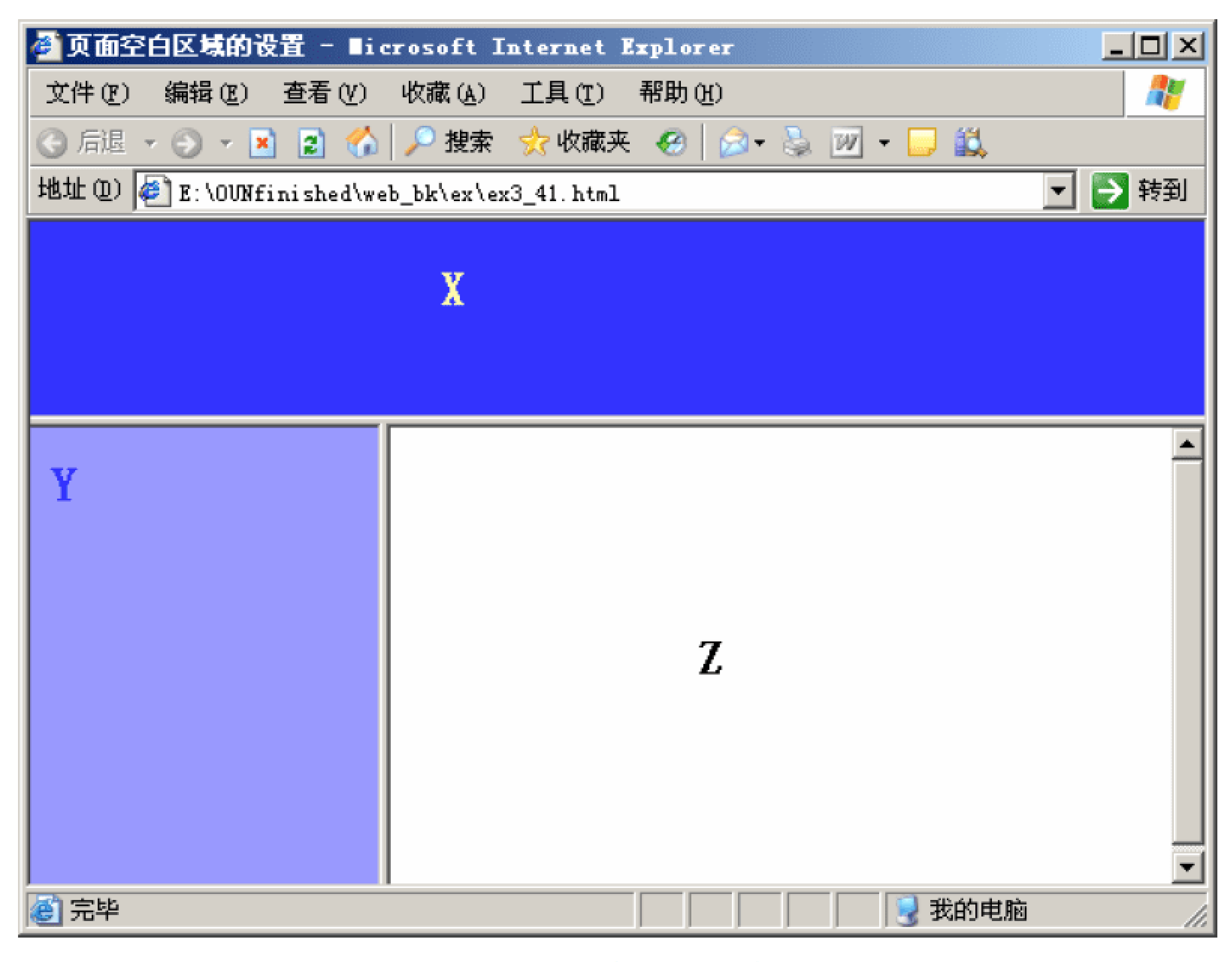


图 4-14　页面空白区域的设置

3. 框架间的链接

在很多网页中，常在一个框架中显示一个所有网页内容的目录，而通过单击其中的某项，在另一个框架中显示相应内容。这些目录是热点文本，需要在框架之间建立超链接，并指明显示的目标文件的框架。

其实只要使用<A>的 target 属性就可以控制目标文件在哪个框架内显示。当单击热点文本时，目标文件就会出现在有 target 指定的框架内。target 属性的值可以为框架名，使用

格式为：

```
<A  href="目标文件名.html"  target="框架名"> 热点文本 </A>
```

框架名有四个特殊的值，分别为_blank、_self、_top 和_parent，可以实现 4 类特殊的操作。

- “_blank”在新的没有名字的浏览器窗口中打开，其定义格式为：

```
<A  href=URL  target=_blank> 热点文本 </A>
```

链接的目标文件被载入一个新的没有名字的浏览器窗口中。

- “_self”在当前的框架打开，其定义格式为：

```
<A  href=URL  target=_self> 热点文本 </A>
```

链接的目标文件被载入当前框架窗口中，代替正在显示的热点文本所在的那个文件。

- “_top”在整个浏览器窗口中打开，其定义格式为：

```
<A  href=URL  target=_top> 热点文本 </A>
```

链接的目标文件被载入整个浏览器窗口中。

- “_parent”在父窗口中打开，其定义格式为：

```
<A  href=URL  target=_parent> 热点文本 </A>
```

当框架有嵌套时，链接的目标文件被载入父框架中。否则，被载入整个浏览器窗口中。

【实例 4-5】框架间的链接

下面的例子说明了在框架间建立链接的不同用法，程序代码如 ex4_5_1.html、ex4_5_2.html 及 Xcol.html 所示。

ex4_5_1.html

```
<HTML>
  <TITLE>框架间的链接 1</TITLE>
  <FRAMESET rows=30%,*>
    <FRAME src="ex4_5_2.html">
      <FRAMESET cols=30%,*>
        <FRAME src="ex4_5_2.html">
        <FRAME src="ex4_5_2.html">
      </FRAMESET>
  </FRAMESET>
</HTML>
```

ex4_5_2.html

```
<HTML>
  <TITLE>框架间的链接 2</TITLE> <NOBR>
  <UL>
  <LI>   &lt;a href="ex4_5_1.html" target="_blank"&gt;<BR>
  <A href="ex4_5_1.html"   target="_blank">单击此处将打开一个新的浏览器窗口来显示这 3 个框架
```

```
</A>&lt;/a&gt;   <P>
        <LI>    &lt;a href="ex4_5_1.html" target="_self"&gt;<BR>
        <A href="ex4_5_1.html" target="_self">单击此处将在本窗口内来显示这 3 个框架</A>&lt;/a&gt;
<P>
        <LI> &lt;a href="xcol.html" target="_parent"&gt;<BR>
        <A href="xcol.html"    target="_parent">单击此处将在这 3 个框架的父窗口(FRAMESET)内来显示 A
窗口</A>&lt;/a&gt; <P>
        <LI> &lt;a href="xcol.html" target="_top"&gt;<BR>
        <A href="xcol.html" target="_top">单击此处将用整个浏览器窗口来显示 A 窗口</A>&lt;/a&gt;
        </UL>
      </HTML>
```

Xcol.html

```
      <HTML>
        <TITLE>X Color</TITLE>
        <BODY bgcolor=3333ff text=ffffaa>
        <H2>X</H2>
      </HTML>
```

运行后的浏览器显示如图 4-15 所示，图中可以看到框架中的不同种类的链接，单击后会出现不同的效果，请读者仔细测试并体会其中的区别。其中左边的图是刚打开文件 ex4_5_1.html 之后的界面，而右边的图所显示的是多次单击“单击此处将打开一个新的浏览器窗口来显示这 3 个框架”后的显示效果。

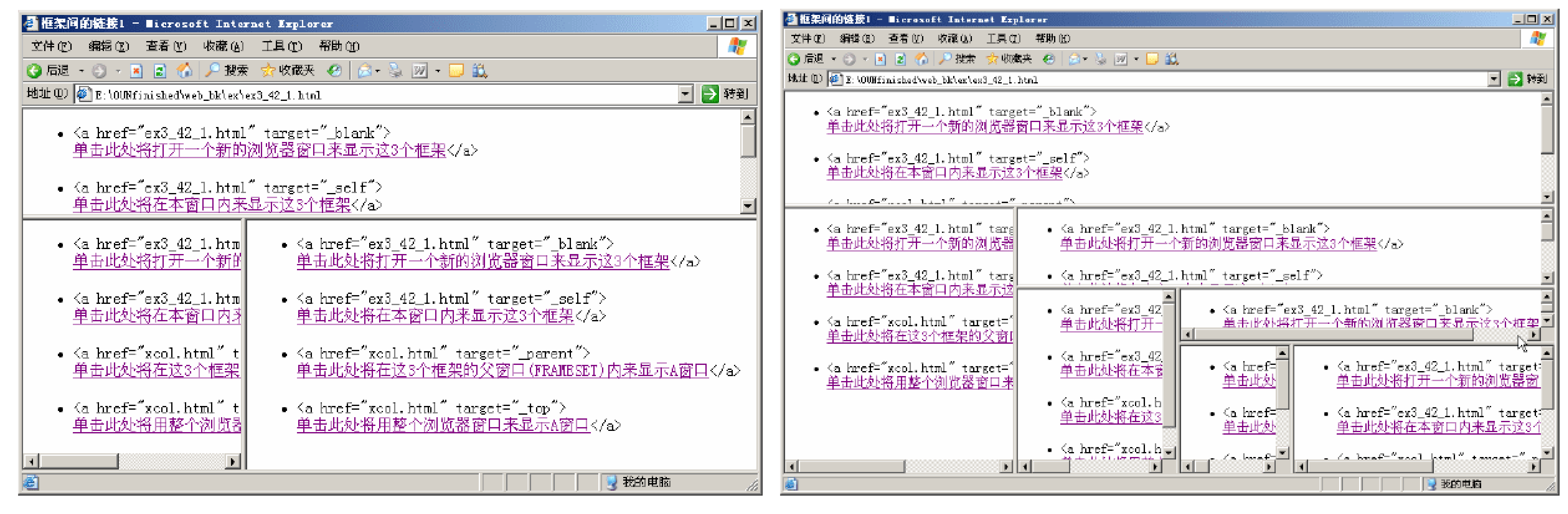


图 4-15　在框架间建立链接

4. DIV+CSS 实现框架的用法

由于 HTML5 中不再支持 frame 的用法，因此，在此开发环境下可以采用一种 DIV 标签配合 CSS 方案的替代用法。

【实例 4-6】采用 DIV+CSS 方案的框架

下面的例子实现了类似于 frame 框架的效果，采用了 DIV+CSS 标签的用法，程序代码如 ex4_6.html 所示。

ex4_6.html

```
      <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
    <title>左侧固定，上方和右侧自适应宽度变化</title>

    <style>
      body {font-family:Verdana; font-size:14px; margin:0;}
      #container {margin:0 auto; width:100%;}
      #header { height:100px; background:#9c6; margin-bottom:5px;}
      #mainContent { height:500px; margin-bottom:5px;}
      #sidebar { float:left; width:200px; height:500px; background:#cf9;}
      #content { margin-left:205px !important; margin-left:202px; height:500px; background:#ffa;}
    </style>
  </head>
  <body>
    <div id="container">
      <div id="header">这是上部区域</div>
      <div id="mainContent">
        <div id="sidebar">这是左侧区域</div>
        <div id="content">2 列左侧固定，右侧自适应宽度+头部。可改变浏览器的宽度来进行测试。
        </div>
      </div>
    </div>
  </body>
</html>
```

本实例中定义了一个 container，其中包含了 header 和 mainContent，在 mainContent 中定义了 sidebar 和 content，在<head>的<style>部分定义了每个部分的相对位置和是否能自适应变化，在网页中的显示效果如图 4-16 所示。

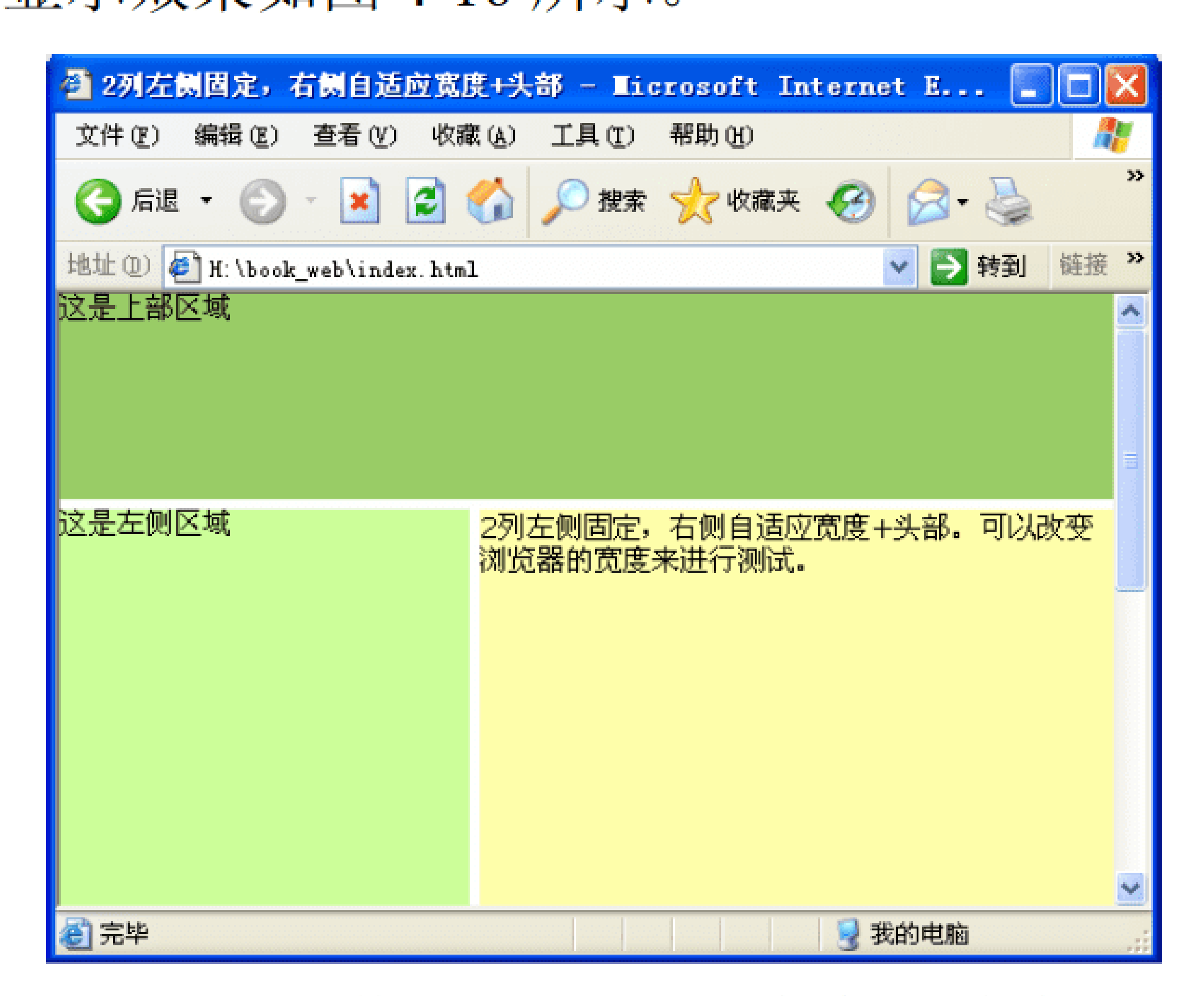


图 4-16　使用 DIV+CSS 方案建立的框架

注意：

本例中使用了第 5 章中讲述的 CSS 技术，对此问题如暂时不了解可以等学习了 CSS 相关章节后再阅读。

另外，当本例中使用 content 设定高度后，像素会跑到 content 外侧，可采用!important 修正在 IE 下向左多浮动像素，这个偏移的差正好是 3 个像素，经过修正后在火狐和 IE 下

的显示就是一样的了。

5. HTML 5 中的 iframe 框架应用

iframe 元素会创建包含另外一个文档的内联框架(即行内框架)。一个典型的 iframe 用法如下：

```
<iframe src="http://www.njupt.edu.cn/" width="400" height="200" scrolling="yes" />
```

其中常用属性及其说明如表 4-4 所示。

表 4-4　<iframe>标签常用属性及其说明

属 性 名 称	说　明
src	表示该框架对应的源文件
scrolling	设置是(yes) / 否 no / 自动(auto)加入滚动条
width	宽度
height	高度
frameborder	规定是否显示框架周围的边框，可设置为 0 或 1
name	规定 iframe 的名称

特别需要说明的是，如果希望隐藏 iframe 出现的滚动条，可以使用 Scrolling=no 来进行操作，也可以使用第 5 章将要介绍的 CSS 样式来隐藏滚动条。当然，如果设置了合适的高宽也可以达到不显示滚动条的目的。

【实例 4-7】采用 iframe 方案的框架用法

下面的例子给出了为达到与 frame 类似效果，而采用 iframe 标签的用法，程序代码如 ex4_7.html 所示。

ex4_7.html

```
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
    <html xmlns="http://www.w3.org/1999/xhtml">
      <head>
        <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
        <title>左侧固定，上方和右侧自适应宽度变化</title>
      </head>
    <body>
    <table width="100%" border="0" cellspacing="0" cellpadding="0">
      <tr>
        <td ><iframe name=top marginwidth=0 marginheight=0 src="http://www.sohu.com" frameborder=0
scrolling="no" width="100%" height="100"></iframe></td>
      </tr>
      <tr>
        <td><table width="100%" border="0" cellspacing="0" cellpadding="0">
          <tr>
            <td width="20%"><iframe name=left marginwidth=0 marginheight=0
```

```
src="http://www.sina.com" frameborder=0 scrolling="no" width="100%" height="500"></iframe></td>
                <td width="80%"><iframe name=body marginwidth=0 marginheight=0
src="http://www.163.com" frameborder=0 scrolling="no" width="100%" height="500"></iframe></td>
              </tr>
            </table></td>
          </tr>
          <tr>
            <td><iframe name=foot marginwidth=0 marginheight=0 src="www.baidu.com" frameborder=0
scrolling="no" width="100%" height="50"></iframe></td>
          </tr>
        </table>
      </body>
```

本实例中使用了 table 元素来控制页面布局，使用 iframe 元素引入了不同网页，在浏览器中的显示效果如图 4-17 所示，其中出现了设定的 3 个不同网站。

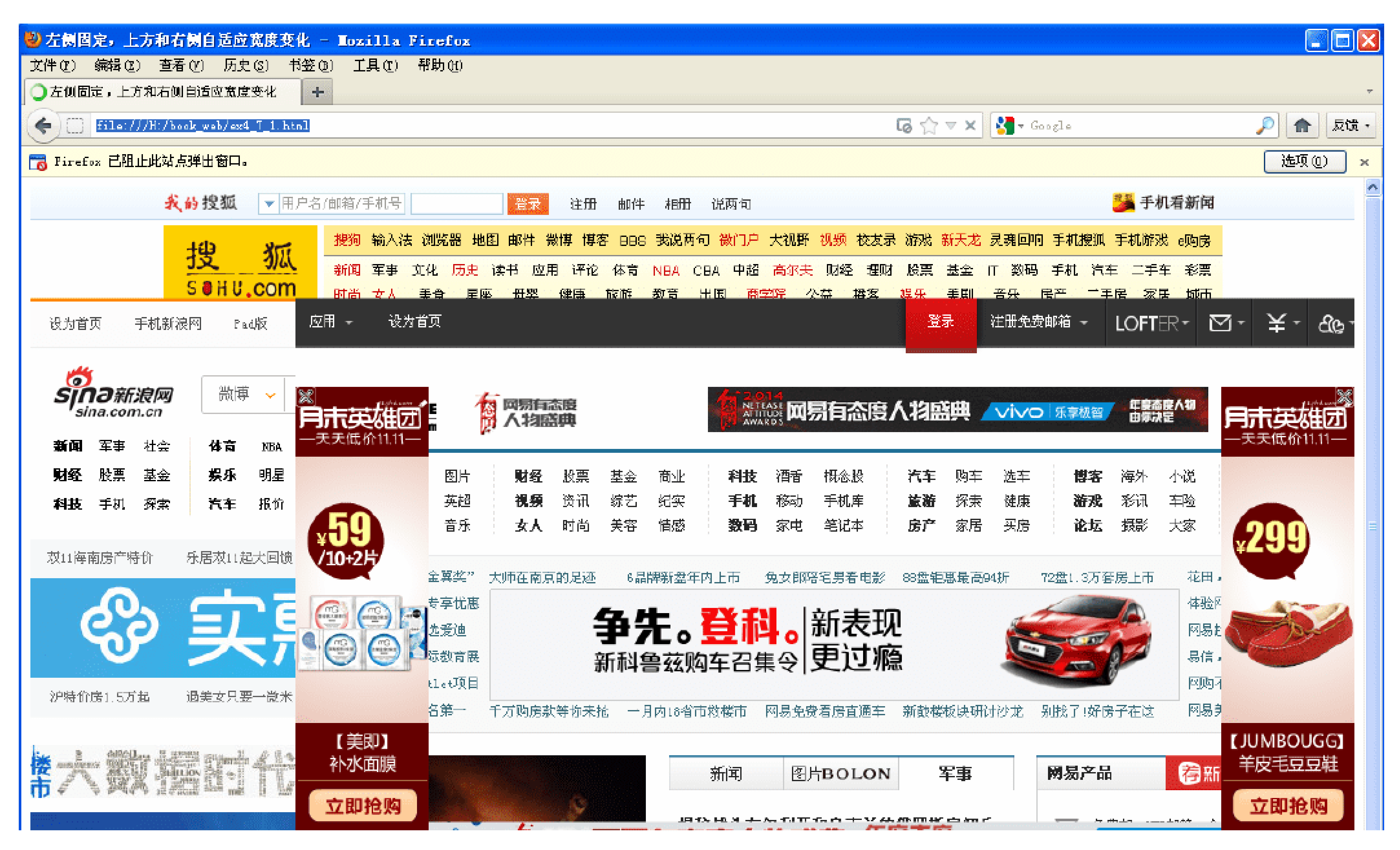


图 4-17　使用 iframe 方案建立的框架

4.2.2　表单

使用表单可以在上一节中介绍的交互功能。网页上具有可输入表项及项目选择等控制所组成的栏目称为表单。在网页中，可以通过表单交流和反馈信息达到与用户交互的目的，与表单有关的标签有<FORM>和<INPUT>，表单的基本语法及格式为：

```
<FORM action="mailto:mail 地址或网址" method=get|post>
  <INPUT type="表项名" name="名" size=x maxlength=y>
…
</FORM>
```

<FORM>标签主要处理表单结果的处理和传送。其中各属性表示的含义如下。

- action 属性：表单处理的方式，往往是 E-mail 地址或网址。
- method 属性：表单数据的传送方向，是获得(GET)表单还是送出(POST)表单。

<INPUT>标签主要用来设计表单中提供给用户的输入形式，对其中的属性说明如下。

- type 属性：指定要加入表单项目的类型(text、password、checkbox、radio、image、hidden、submit、reset)。
- name 属性：该表项的控制名，主要在处理表单时起作用。
- size 属性：单行文本区域的宽度。
- maxlength 属性：允许输入的最大字符数目。

1. 文字和密码的输入

使用<INPUT>标签的 type 属性，可以在表单中加入表项，并控制表项的风格。type 属性值为 text，则输入的文本以标准的字符显示；type 属性值为 password，则输入的文本显示为“*”。

在表项前应加入表项的名称，如“您的姓名”等，以告诉用户在随后的表项中输入的内容。

【实例 4-8】文字和密码的输入

下面的例子说明了利用表单来输入文字和密码的用法，程序代码如 ex4_8.html 所示。

ex4_8.html

```
<HTML>
  <HEAD><TITLE>输入文本和密码</TITLE><HEAD>
  <BODY text=blue>
    <CENTER><H2><FONT color=red>个人资料</FONT></H2></CENTER>
    <FORM action="mailto:YourMailAdd@YourMail.com" memethod=POST>
      姓名: <INPUT type=text name=姓名><BR>
      主页的网址: <INPUT type=text name=网址  value=http://><BR>
      密码: <INPUT type=password name=密码><BR>
      <INPUT type=submit value="发送"> <INPUT type=reset value="重设">
    </FORM>
  </BODY>
</HTML>
```

运行后的浏览器显示如图 4-18 所示，图中可以看到在文本框中输入的“姓名”、“主页的网址”中可以进行文字的输入和显示；而在密码的输入框中输入后显示出的为圆点。



图 4-18　文字和密码的输入

2. 重置和提交

如果用户想清除输入到表单中的全部内容，可以使用<INPUT>标签中的 type 属性所设的重置(reset)按钮，当表单的条目较多时可以省去在重新输入之前，一项一项删除的麻烦。当用户完成表单的填写后欲发送时，可使用<INPUT>标签属性中 type 所设的提交(submit)按钮，将表单内容发送给 action 中的网址或

函件信箱。定义的一般格式为：

```
<INPUT type="reset" value="按钮名">
<INPUT type="submit" value="按钮名">
```

当默认 value 的设置值时，重置和提交的按钮分别显示为“重置”和“提交查询内容”。例如：

```
<FORM action="mailto:YourMailAdd@YourMail.com" memethod=POST>
  <INPUT type=text name=a01 size=40><BR>
  <INPUT type=text name=a02 maxlength=5><BR>
  <INPUT type=submit><INPUT type=reset>
</FORM>
```

3. 复选框和单选按钮

在页面中有些地方需要列出几个项目，让用户通过选择按钮来选择项目。选择按钮可以是复选框(checkbox)或单选按钮(radio)。用<INPUT>标签的 type 属性可设置选择按钮的类型，属性 value 可设置该选择按钮的控制初值，用以告诉表单制作者选择结果。用 checked 表示是否为默认选中项。name 属性是控制名，同一组的选择按钮的控制名是一样的。

【实例 4-9】复选框和单选按钮的使用

下面的例子说明了在表单中使用复选框和单选按钮来完成输入的用法，程序代码如 ex4_9.html 所示。

ex4_9.html

```
<HTML>
  <HEAD><TITLE>个人资料</TITLE><HEAD>
  <BODY text=green>
    <FORM action="mailto:YourMailAdd@YourMail.com" memethod=POST>
      <CENTER><H2><FONT color=purple>个人资料</FONT></H2></CENTER>
      姓名: <INPUT type=text name="xm" size=12><BR>
      性别: <INPUT type=radio name="性别" value="男" checked>男
           <INPUT type=radio name="性别" value="女">女<BR>
      出生日期: <INPUT type=text name="year" size=2>年
        <INPUT type=text name="month" size=2>月
        <INPUT type=text name="day" size=2>日<BR>
      个人爱好:<INPUT type=checkbox name="爱好" value="体育">体育
        <INPUT type=checkbox name="爱好" value="文学">文学
        <INPUT type=checkbox name="爱好" value="艺术">艺术
        <INPUT type=checkbox name="爱好" value="旅游">旅游
        <INPUT type=checkbox name="爱好" value="美食">美食
        <INPUT type=checkbox name="爱好" value="其他">其他<BR>
    </FORM>
  </BODY>
</HTML>
```

运行后的浏览器显示如图 4-19 所示，从图中可以看到在“性别”、“个人爱好”两项

中分别使用了单选按钮和复选框，用户使用时可以直接使用鼠标单击，而无须用键盘来输入。

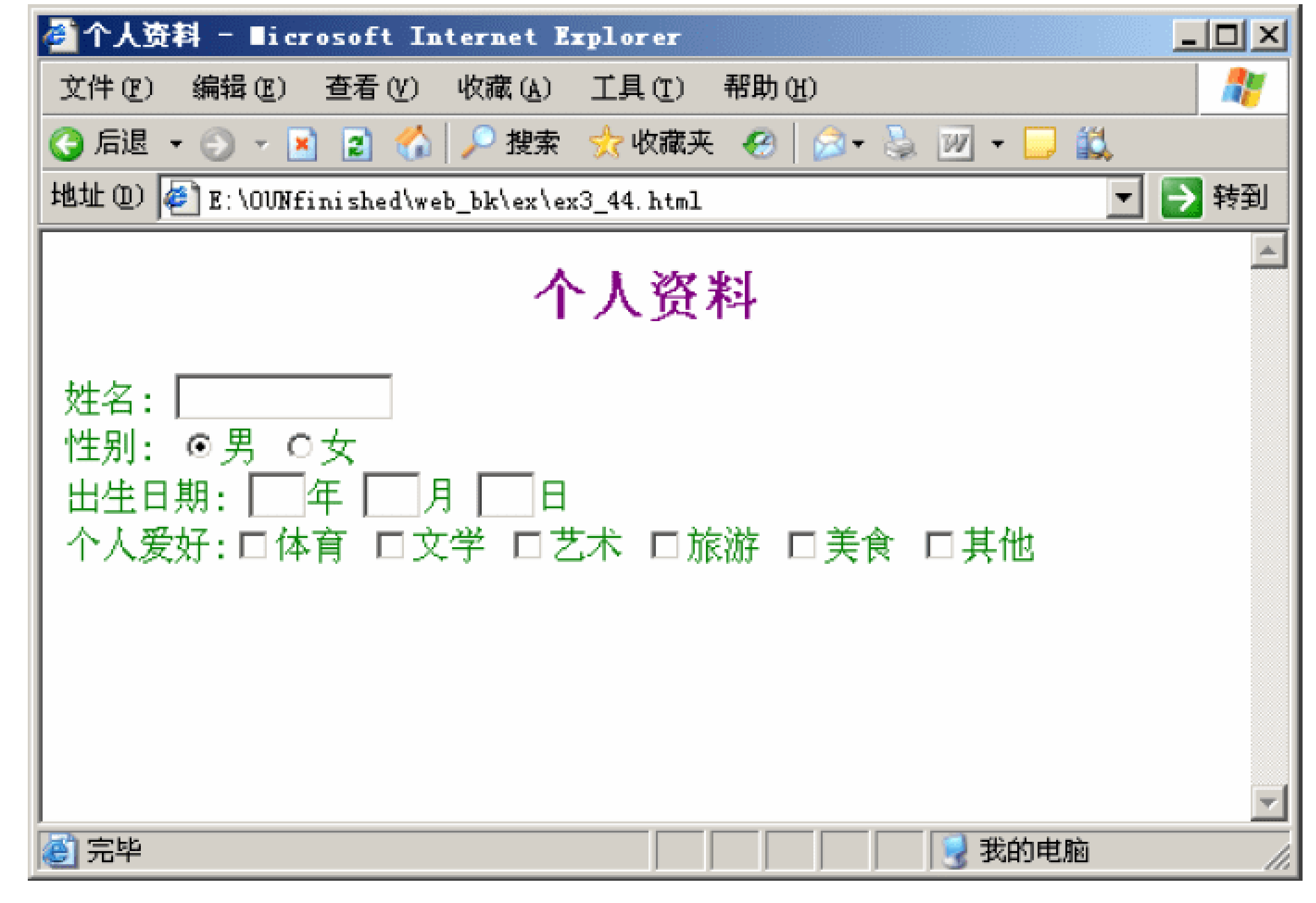


图 4-19　复选框和单选按钮的使用

4. 选择栏

当用户选择的项目较多时，如果用选择按钮来选择，占页面的区域就会较多。可以用<SELECT>标签和<OPTION>标签来设置选择栏。选择栏可分为两种，即弹出式和字段式。<SELECT>标签的格式为：

```
<SELECT size=x name="控制操作名" multiple>
   <OPTION ...>
   ...
</SELECT>
```

其中选择栏中<SELECT>标签所使用的各个属性及其说明如表 4-5 所示。

表 4-5　<SELECT>标签各个属性及其说明

属　性	说　明
size	取数字，表示在带滚动条的选择栏中一次可见的列表项数
name	控制操作名
multiple	不带值，加上本项表示可选多个选项，否则只能单选

<OPTION>标签的格式为：

```
<OPTION select value="可选择的内容">
```

其中选择栏中<OPTION>标签所使用的各个属性及其说明如表 4-6 所示。

表 4-6　<OPTION>标签各个属性及其说明

属　性	说　明
select	不带值，加上本项表示该项是预置的
value	指定控制操作的初始值，默认时初值为 option 中的内容表示选项值

选择栏可以使用弹出式或字段式，说明如下。

(1) 弹出式选择栏

弹出式选择栏的格式为：

```
<FORM>
    <SELECT>
      <OPTION>选项 1
      <OPTION>选项 2
        …
      <OPTION>选项 n
    </SELECT>
</FORM>
```

其中第 1 个选项将作为默认设置。

(2) 字段式选择栏

字段式选择栏与弹出式选择栏的主要区别在于在<SELECT>中的 size 属性值取大于 1 的值，此值表示在选择栏中不拖动滚动条可以显示选项的数目。

【实例 4-10】选择栏的使用

下面的例子说明了在表单中使用两种不同的选择栏的方法，程序代码如 ex4_10.html 所示。

ex4_10.html

```
<HTML>
  <HEAD> <TITLE>请填写个人资料</TITLE> <HEAD>
  <BODY text=blue>
    <FORM action="mailto:YourMailAdd@YourMail.com" method=POST>
      <CENTER><H3><FONT color=blue>请填写个人资料</FONT></H3></CENTER>
      姓    名: <INPUT type=text name="xm" size=12><BR>
      性    别: <INPUT type=radio name="性别" value="男" checked>男
      <INPUT type=radio name="性别"   value="女">女<BR>
      出生日期: <INPUT type=text name="year" size=2>年
      <INPUT type=text name="month" size=2>月
      <INPUT type=text name="day" size=2>日<BR>
      <TABLE>
        <TR><TD>个人爱好:<TD><INPUT type=checkbox name="爱好" value="体育">体育<TD>
        <INPUT type=checkbox name="爱好" value="文学">文学<TD>
        <INPUT type=checkbox name="爱好" value="艺术">艺术<TD>
        <INPUT type=checkbox name="爱好" value="旅游">旅游<TD>
        <TR><TD><TD><INPUT type=checkbox name="爱好" value="美食">美食<TD>
        <INPUT type=checkbox name="爱好" value="其他">其他</TABLE><BR>
     学历: <SELECT name="学历" size=1>
        <OPTION value="中专">中专
        <OPTION selected value="大专">大专
        <OPTION value="大学">大学
        <OPTION value="硕士">硕士
        <OPTION value="博士">博士
        <OPTION value="其他">其他
      </SELECT>   
```

```
            职称:<SELECT name="职称" size=3>
              <OPTION value="助教">助教
              <OPTION value="讲师">讲师
              <OPTION value="副教授">副教授
              <OPTION value="教授">教授
              <OPTION value="其他">其他
            </SELECT><P>
            <CENTER><INPUT type=submit value="提交
"> <INPUT type=reset value="重填"></CENTER>
          </FORM>
        </BODY>
    </HTML>
```

运行后的浏览器显示如图 4-20 所示，从图中可以看到在“学历”、“职称”两项中分别使用了弹出式选择栏和字段式选择栏。

图 4-20　选择栏的使用

5. 多行文本输入框

在意见反馈栏中往往需要用户发表意见和建议，由于可能会录入较多文字，因此提供的输入区域一般较大。为此可以使用<TEXTAREA>标签，它可以设置成允许多行文本的输入，设置的格式为：

```
<TEXTAREA name="控制操作名" rows="行数" cols="列数">
    多行文本
</TEXTAREA>
```

其中的行数和列数是指不用滚动条就可看到的部分。

【实例 4-11】多行文本输入框的使用

下面的例子说明了在表单中使用多行文本输入框的方法，程序代码如 ex4_11.html 所示。

ex4_11.html

```
<HTML>
  <HEAD><TITLE>请填写个人资料</TITLE><HEAD>
  <BODY text=blue>
    <FORM action=/cgi-bin/post-information method=POST>
      <CENTER><H2><FONT color=blue>请填写个人资料</FONT></H2></CENTER>
      姓名:<INPUT type=text name="xm" size=12>  
      性别:<INPUT type=radio name="性别" value="男" checked>男
        <INPUT type=radio name="性别"    value="女">女  
      出生日期:<INPUT type=text name="year" size=2>年
      <INPUT type=text name="month" size=2>月
      <INPUT type=text name="day" size=2>日<BR>
      个人爱好:<INPUT type=checkbox name="爱好" value="体育">体育
        <INPUT type=checkbox name="爱好" value="文学">文学
        <INPUT type=checkbox name="爱好" value="艺术">艺术
        <INPUT type=checkbox name="爱好" value="旅游">旅游
        <INPUT type=checkbox name="爱好" value="美食">美食
```

```
        <INPUT type=checkbox name="爱好" value="其他">其他<BR>
      学历:<SELECT name="学历" size=1>
        <OPTION value="中专">中专
        <OPTION selected value="大专">大专
        <OPTION value="大学">大学
        <OPTION value="硕士">硕士
        <OPTION value="博士">博士
        <OPTION value="其他">其他
      </SELECT>   
      职称:<SELECT name="职称" size=3>
        <OPTION value="助教">助教
        <OPTION value="讲师">讲师
        <OPTION value="副教授">副教授
        <OPTION value="教授">教授
        <OPTION value="其他">其他
      </SELECT><BR>
      个人简历:<TEXTAREA name=comment rows=5 cols=60>
        </TEXTAREA><BR><BR>
        <CENTER><INPUT type=submit value="提交">
        <INPUT type=reset value="重填"></CENTER>
     </FORM>
   </BODY>
</HTML>
```

运行后的浏览器显示如图 4-21 所示，从图中可以看到在表单中的“个人简历”项为多行文本输入框，其中可以填写多行文本内容。

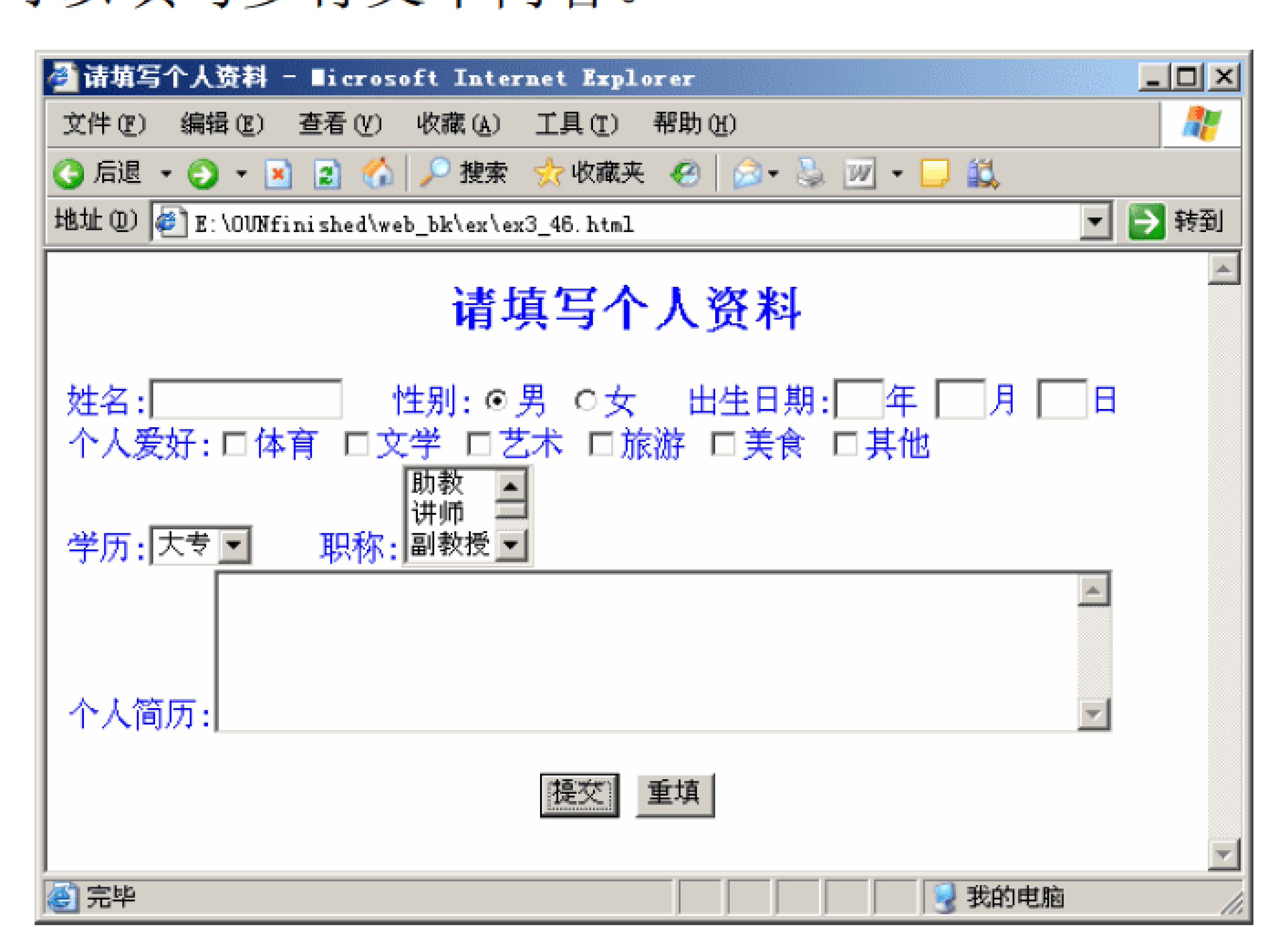


图 4-21　多行文本输入框的使用

注意：

限于篇幅，这里仅说明了在表单中常见的五种元素，更多种类的元素请读者参考更详细的参考手册。

6. 表单设计技巧

Web 应用程序总是利用表单来处理数据录入和配置，但并不是所有的表单都保持一致。在《HTML 权威指南》一书中，作者提出：“输入框(Input)应当符合逻辑地划分为小

组，这样大脑就可以很好地处理大堆区域间的关系。”输入区域的对齐方式、各自的标签(label)、操作方式、以及周围的视觉元素都会或多或少地影响用户的行为。

(1) 垂直排列

考虑到用户完成表单填写的时间应当尽可能的短，并且收集的数据都是用户所熟悉的(比如姓名、地址、付费信息等)，垂直对齐的标签和输入框可以说是最佳的。每对标签和输入框垂直对齐给人一种两者接近的感觉，并且一致的左对齐减少了眼睛移动和处理的时间。用户只需要往一个方向移动几下。在此读者可以回忆自己的 Web 体验，实际上很多网站的注册页面都是按照这种方式来组织的，如图 4-22 所示。



图 4-22　垂直排列的表单

在这种布局中，推荐使用加粗的标签，这可以增加它们的视觉比重，提高其显著性。如不加粗的话，从用户的角度看，标签和输入框的文字几乎就一样了。

(2) 关于对齐

如果一个表单上的数据并不为人熟悉或者在逻辑上分组有困难(比如一个地址的多个组成部分)，左对齐的标签可以很容易地通览表单的信息。用户只需要上下看看左侧一栏标签就可以了，而不会被输入框打断思路。但这样一来，标签与其对应的输入框之间的距离通常会被更长的标签拉长，可能会影响填写表单的时间。用户必须左右来回地跳转目光来找到两个对应的标签和输入框。

于是产生了一种替代的方案，标签右对齐布局，使得标签和输入框之间的联系更紧密。然而结果是左边参差不齐的空白和标签让用户很难快速检索表单要填写的内容。在西方国家，人们习惯于从左至右书写，所以这种右对齐的布局就给用户造成了阅读障碍。

(3) 改善用户体验：增加视觉元素

由于“标签左对齐布局”的优点(方便检索并且减少垂直高度)，尝试纠正它的主要缺点(标签和输入框的分离)就很诱惑人。一个方案就是增加背景色和分割线，不同的背景色产生了一列垂直的标签和一列垂直的输入框，每一组标签和输入框利用清晰的水平线分开。虽然这听上去不错，但是这样做仍然存在问题。

对比之前的形态(用户主观的视觉区分)，如果增加了中间线、一个个有背景色的单元格以及一条条的水平线，这些元素会转移用户的视线，让用户难以聚焦到一些重要的元素上，比如标签和输入框。正如 Edward Tufte 指出的：“信息本身存在差异，必然产生感官上的不同。”换句话说，任何对布局无用的视觉元素都会不断地扰乱布局。当试着浏览左侧的标签时就可以发现，视线总是被打断，被迫停下来想那些水平线、单元格和背景颜色。

当然这并不意味着放弃背景色和线条。它们对于划分相关区域信息还是很有效的。比如一条细水平线或者一个浅浅的背景色，都可以从视觉上组合相关数据。背景色和线条对于区分表单的主要操作按钮尤其有效。

一个表单的主要操作(通常是“提交”或“保存”)需要一个比较强的视觉比重(如采用亮色调、粗字体、背景色等)。这给予用户一个暗示：您已/即将完成填写表单。当一个表单有多个操作，比如“继续”和“返回”，就有必要减轻次要操作的视觉重量。这可以最小化用户潜在的操作错误的风险。

上述的原则可以帮助表单的设计，但是组合布局、可视化元素以及内容，仍然需要经过更完整的测试以及分析才能真正符合用户的需求，更多的内容读者可以参考有关网页视觉设计的专门讨论。

4.2.3　脚本

脚本可以分为服务器端脚本和客户端脚本，在 HTML 代码中嵌入的脚本属于客户端脚本。

HTML 规范中常用的标签非常简单，这是 HTML 语言的一个特点，但同时也是其功能受到很大限制的原因。如果希望浏览器能响应用户的要求，浏览器必须具备处理事件的能力，也就是说，事件是浏览器响应用户交互操作的一种机制。当然，任何程序包括浏览器本身都有一套已经设计好的响应各种事件的方法。脚本的事件处理机制可以改变浏览器响应用户操作的标准方法，这样就可以开发出更加具有交互性，更容易使用的 Web 页面。利用脚本的事件，主要有下面的两个用途：

- 验证用户输入窗体的数据；
- 增加页面的动态效果。

注意：

本书的第 6 章 JavaScript 语言中专门介绍了这部分知识，此处就不再讨论，请读者查阅该章。

4.2.4　网页中加入动态效果和多媒体

HTML 考虑了在网页中生成动态效果和多媒体，这里将从滚动字幕、网页中加入视频和音频等方面进行介绍。

1. 滚动字幕

我们常常可以看到网页上的滚动字幕，实际上制作滚动字幕不需要复杂的技巧，使用 HTML 规范中的<MARQUEE>标签即可。<MARQUEE>标签的格式为：

```
<MARQUEE align="top/middle/bottom" bgcolor="颜色值" width=x 或 x% height=y
direction="left/right" loop=-i/-1/infinite behavior="scroll/side/alternate" hspace=m vspace=n scrollamount=数值
scrolldelay=数值>需要滚动的文字</MARQUEE>
```

<MARQUEE>标签的属性如表 4-7 所示。

表 4-7　<MARQUEE>标签的属性及其说明

属　　性	说　　明
align	设置字幕和垂直文本对齐
bgcolor	设置字幕的背景色
width	设置字幕的宽，x 为像素数或相对于窗口宽的百分比
height	设置字幕的高，y 为像素数
direction	设置字母的方向，可以取 left/right
loop	设置字幕的循环次数 i，当为-1 或 infinite 时为无限循环
behavior	scroll 设置文字单向流动，side 设置流动文字到达边界停止，alternate 设置流动文字到达边界后反向流动
hspace	水平方向空白像素数
vspace	垂直方向空白像素数
scrollamount	字母移动速度
scrolldelay	移动每步的延时

【实例 4-12】滚动字幕的使用

下面的例子说明了滚动字幕的用法，程序代码如 ex4_12.html 所示。

ex4_12.html

```
<html>
  <head><title>滚动字幕</title>
  </head>
  <body>
    <MARQUEE behavior=alternate width=50%>
      <IMG SRC="cartoon.gif" border=0>
      <FONT color=#0bcdef>图片和文字都可以动！</FONT>
    </MARQUEE>
    <MARQUEE direction=left>我往左跑</MARQUEE><P>
    <MARQUEE direction=right>我往右跑</MARQUEE><P>
  </body>
</html>
```

运行后的浏览器显示如图 4-23，图中的文字从屏幕的一端移动至另一端，可以移动的不仅仅是文字，图片也可以同时移动，读者可以运行这段代码来观察。

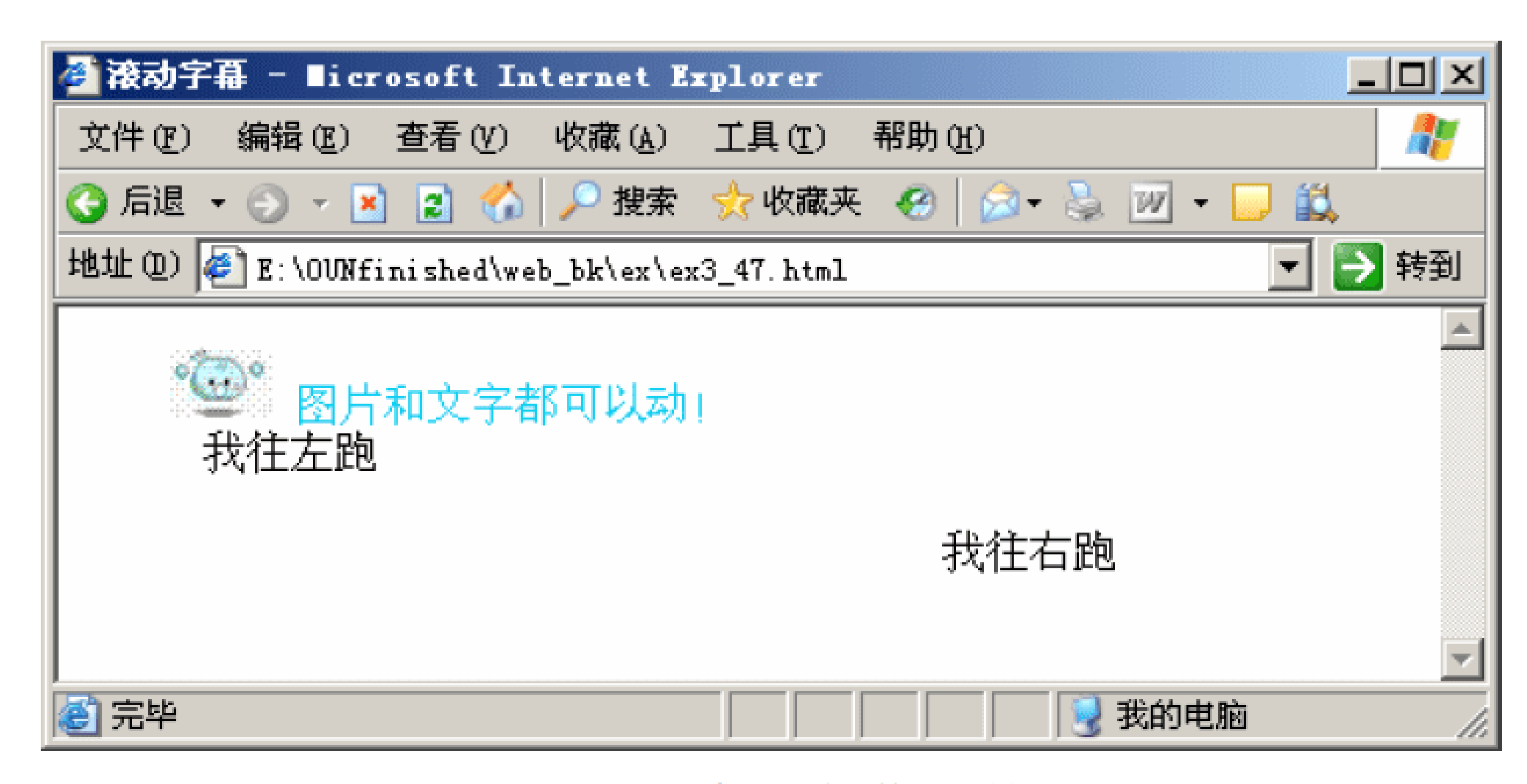


图 4-23　滚动字幕的使用

2. 网页中加入音频和视频

(1) 加入音频

我们可以使用<BGSOUND>标签在网页中加入背景音乐，定义的格式为：

```
<BGSOUND src="声音文件"  loop="播放次数">
```

要加入的背景音乐文件的格式可以为.wav、.au 或.mid。当设置属性 loop(播放次数)为−1 或 INFINTE 时，声音将一直播放直到关闭该网页为止。

(2) 加入视频

用<IMG>标签的 dynsrc 属性可以向网页中加入.avi 等格式的视频剪辑文件。定义的格式为：

```
<IMG  src="图像文件"  dynsrc="视频剪辑文件.avi" loop="次数" loopdelay="时间" start="值"
controls>
```

其中，dynsrc 用来指定播放视频文件存放的路径和文件名；src 用来指定在浏览器尚未完全读入 AVI 文件时，先在 AVI 播放区域显示的图像。例如：

```
<IMG src="sample.gif" dynsrc="sample.avi">
```

loop 指定视频文件播放的次数，如果值为 infinite 则反复播放直到关闭该网页。

loopdelay 指定两次播放的间隔时间，单位是毫秒。例如：

```
<IMG src="sample.gif" dynsrc="sample.avi" loop=2 loopdelay=10>
```

start 指定何时开始播放视频文件。它的两个属性值为 fileopen(默认)和 mouseover。fileopen 是在链接到含本标签的网页时开始播放，mouseover 是将鼠标移到视频播放区时才开始播放。也可以两者同时设置：<IMG start=fileopen,mouseover>。另外，用鼠标在视频播放区域单击一下，也将使该视频开始播放。例如：

```
<IMG src=" sample-s.gif" dynsrc="sample-s.avi" start=mouseover>
```

controls 用来在视频窗口下附加视频播放控制栏，可以用这个控制工具栏暂停、启动或跳过视频文件中的某一部分。

此外，还有<IMG>标签常见的属性，如 width、height、align 等。

【实例 4-13】在网页中加入视频

下面的例子展示了在网页中加入视频的方法，程序代码如 ex4_13.html 所示。

ex4_13.html

```
<HTML>
  <HEAD><TITLE>插入多媒体文件</TITLE></HEAD>
  <body>
    <H2 ALIGN="CENTER">网页中的多媒体</H2>
    <HR>
    <center>
      <embed src="sample.avi" width="160" height="90" loop="true">
    </center>
```

```
  </BODY>
</HTML>
```

请读者自己运行后查看效果。

3. 自动刷新页面

自动刷新页面就是页面打开停留几秒钟后自动指向其他设定的网页。其格式为：

```
<META http-equiv="Refresh" content="秒数; url=新页面">
```

其中，<META>标识必须放置在<HEAD>...</HEAD>中。

http-equiv 属性值设置为“Refresh”时，要求显示 URL 指定的文件。

content 属性包含两个值：秒数和 URL，它们之间用“;”分隔。该链接将在指定的时间后被打开。

【实例 4-14】页面自动刷新

下面的例子说明了页面自动刷新的用法，程序代码如 ex4_14.html 所示。

ex4_14.html

```
<HTML>
  <HEAD>
    <TITLE>自动刷新页面</TITLE>
    <meta http-equiv="refresh" content="5;url=http://www.sohu.com">
  </HEAD>
  <BODY>
    五秒钟后将自动连向首页；若五秒钟后无法连结。
    请按下列链接，谢谢！
    <a href=http://www.sohu.com>http://www.sohu.com</a>
  </BODY>
</HTML>
```

运行后的浏览器显示如图 4-24 所示，左边的图是该网页最初显示的结果，由于设定的时间为 5 秒钟，5 秒钟后网页转向了设定的 http://www. sohu.com，见图 4-24 右图。

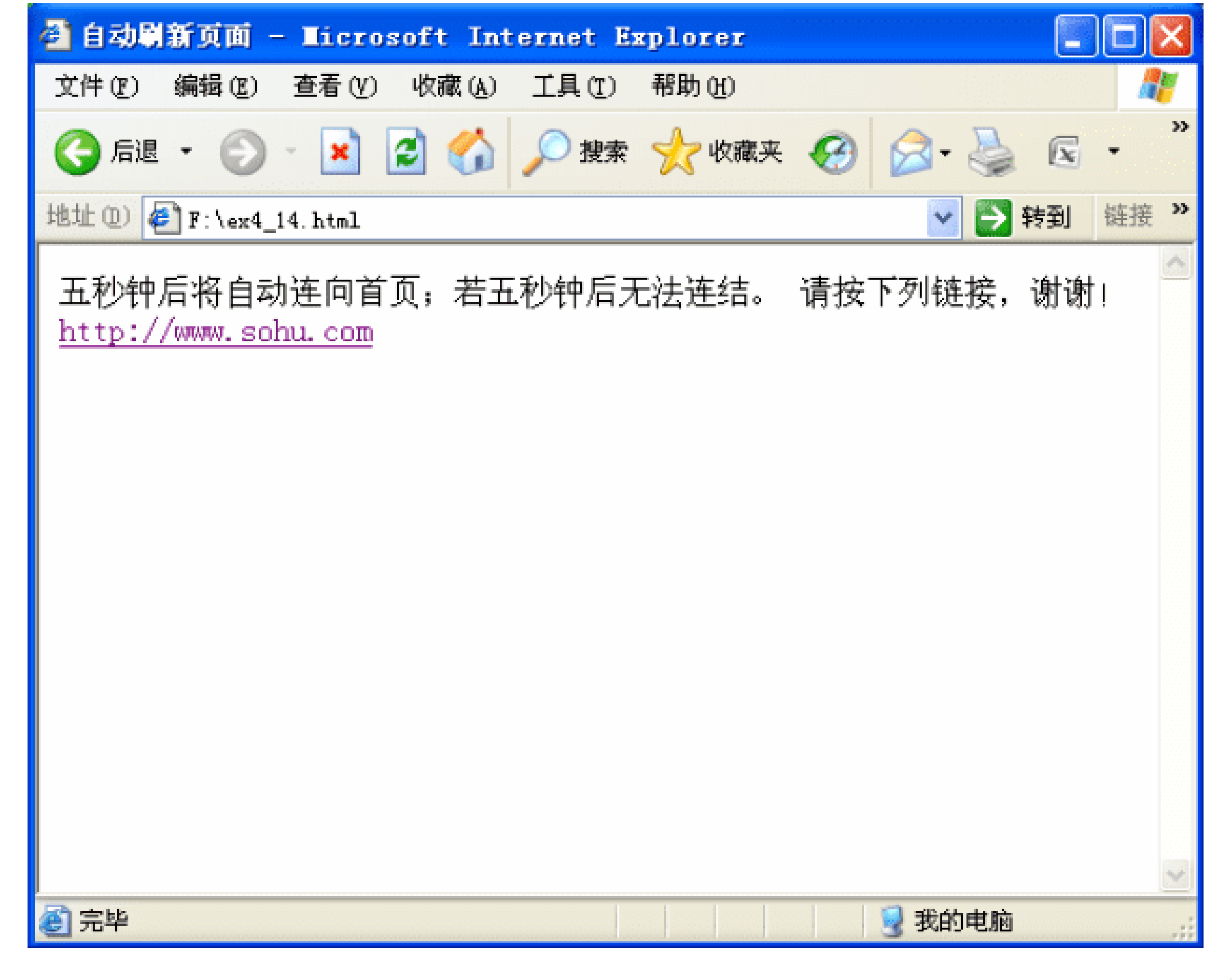




图 4-24　页面自动刷新的使用

4.2.5　可执行对象

HTML 除了可以支持嵌入文本、图像、视频、动画等媒体元素外，还可以嵌入可执行的对象，如 Java Applet、ActiveX 控件、插件等。所嵌入的对象在客户侧运行，并将运行得到的结果直接显示在客户端。利用可执行对象，可以动态地改变 HTML 文档的内容，同时还带来了与用户动态交互的可能性，因此有时将这种文档称为活动文档。

1. Java Applet

Applet 可以翻译为小应用程序，Java Applet 就是用 Java 语言编写的这样的一些小应用程序，它们可以直接嵌入到网页或者其他特定的容器中，并能够产生特殊的效果。

Applet 必须运行于某个特定的“容器”，这个容器可以是浏览器本身，也可以是通过各种插件，或者包括支持 Applet 的移动设备在内的其他各种程序来运行。在运行时 Applet 通常会与用户进行互动，显示动态的画面，并且还会遵循严格的安全检查，阻止潜在的不安全因素(例如根据安全策略，限制 Applet 对客户端文件系统的访问)。

在 Java Applet 中，可以实现图形绘制、字体和颜色控制、动画和声音的插入、人机交互及网络交流等功能。Applet 还提供了利用用户计算机 GUI 元素的功能，因此可以建立标准的图形用户界面，如窗口、按钮、滚动条等。目前，在网络上有非常多的 Applet 范例，读者可以去调阅相应的网页以观看它们的效果。

Applet 依赖于浏览器的调用，它是通过〈Applet〉标签嵌入在 HTML 文件中的，定义的格式如下：

```
<APPLET code="Applet 文件标识" codebase="Applet 文件所在路径" width="Applet 显示区域的宽度" height="Applet 显示区域的高度" name="Applet 的符号名">
</ APPLET>
```

而最简单的调用可以直接写成：

```
<applet code=TicTacToe.class width=120 height=120> </applet>
```

【实例 4-15】将 Java Applet 嵌入网页

下面的例子说明了将 Java Applet 嵌入网页的方法，程序代码如 ex4_15.html 所示。

ex4_15.html

```
<html>
  <head>
    <title>Java Applet 的用法</title>
    </head>
  <body>
      <h1>Java Applet 的用法</h1>
      <hr>
      <applet code=TicTacToe.class width=120 height=120>
alt="Your browser understands the &lt;APPLET&gt; tag but isn't running the applet, for some reason."
Your browser is completely ignoring the &lt;APPLET&gt; tag!
      </applet>
```

```
    </body>
</html>
```

运行后的浏览器显示如图 4-25 所示，在界面上出现了一个简单的游戏，用鼠标单击后不仅可以和计算机对弈，还能听到声音，这些功能均是在 Applet 中进行编写的，网页中只是对 TicTacToe.class 这个程序进行了调用。

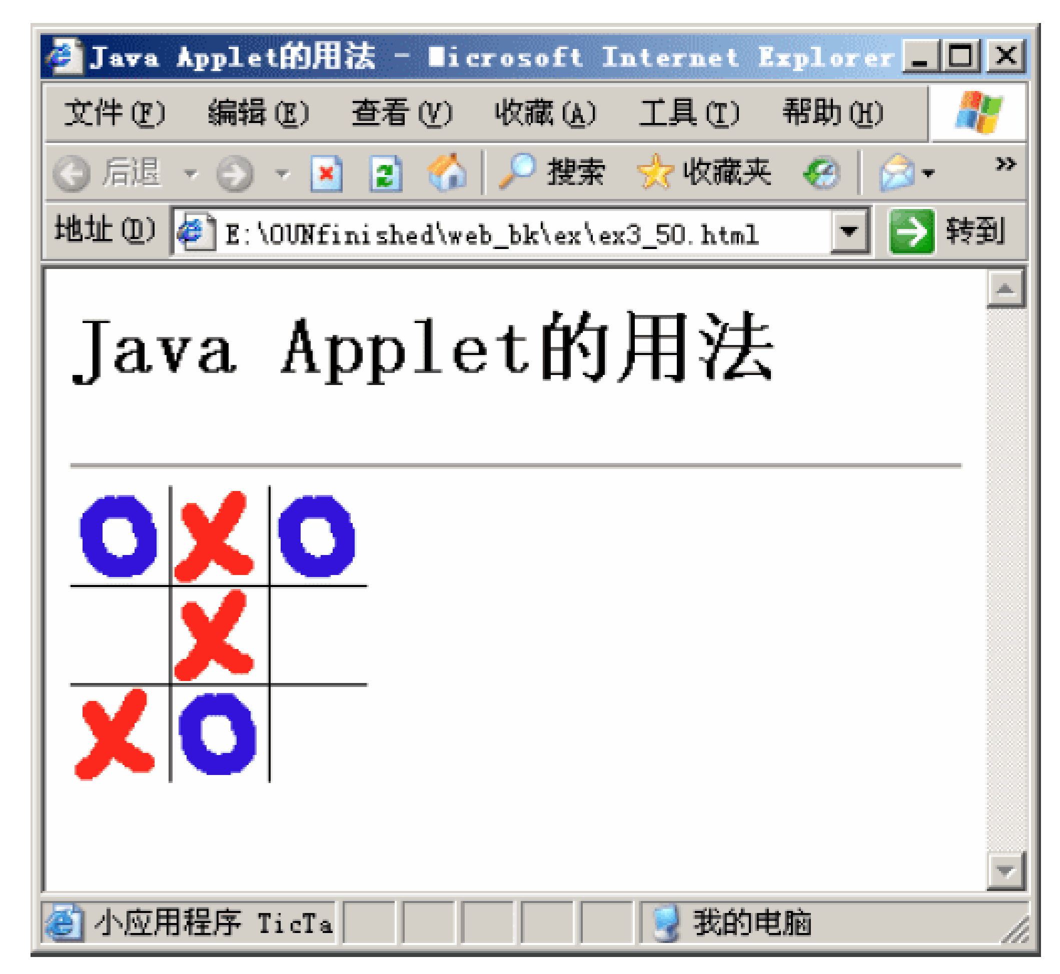


图 4-25　将 Java Applet 嵌入网页

2. Flash

Flash 是当今 Internet 上最流行的动画作品(如网上各种动感网页、LOGO、广告、MTV、游戏和高质量的课件等)制作工具，并成为事实上的交互式矢量动画标准，其影响之大就连软件巨头微软也不得不在其新版的 Internet Explorer 内嵌 Flash 播放器。

由于在 Flash 中采用了矢量作图技术，各元素均为矢量，因此只用少量的数据就可以描述一个复杂的对象，从而大大减少动画文件的大小。而且矢量图像还有一个优点，就是可以真正做到无极放大和缩小，动画的任意缩放都不会有任何失真或锯齿。

Flash 之所以在网上广为流传，其文件小、传输快只是一个方面，还有一点就是采用了流控制技术。简单的说，也就是边下载边播放的技术，不用等整个动画下载完，就已经开始播放了。

Flash 动画还支持过渡变形技术，包括移动变形和形状变形。“过渡变形”方法只需制作出动画序列中的第一帧和最后一帧(关键帧)，中间的过渡帧可通过 Flash 计算自动生成。这样不但可以大大减少动画制作的工作量，缩减动画文件的尺寸，而且过渡效果非常平滑。对帧序列中的关键帧的制作，产生不同的动画和交互效果。播放时，也是以时间线上的帧序列为顺序依次进行的。

Flash 动画与电影的一个显著区别就是其具有交互性。也就是通过使用键盘、鼠标等可以在作品各个部分跳转，使受众参与和控制动画的播放。Flash 交互是通过 Action Script 实现的。Action Script 是 Flash 的脚本语言，随着其版本的不断更新而日趋完美。使用 Action Script 可以控制 Flash 电影中的对象、创建导航和交互元素。

注意：

Flash 对象可以方便地嵌入网页之中，只要使用<object>对象，并对其进行必要的设置即可。

3. 通用对象——OBJECT

考虑到将来可能出现的各种媒体类型或可执行对象，HTML 中引入了 OBJECT 这种通用的对象机制。<OBJECT>标签不仅可以用来在页面中插入 ActiveX 控件，还可以插入其

他的 OLE 对象，如图像、文档、动画、小程序等，对象的定义可出现在文档的<HEAD>或<BODY>中，其格式为：

```
<OBJECT classid="对象的类标识或 URI" width="对象显示区域的宽度" height="对象显示区域的高度" codebase="对象文件所在路径" codetype="可执行对象的类型" data="对象数据的 URI" type="对象数据的类型" >
<PARAM name="属性名称" value="该属性的值">
   …
</OBJECT>
```

它的各种属性分别用来设置所插入对象的类型、路径、在页面中显示的大小、位置等，而从外界传递给对象的参数是由 PARAM 元素提供的。需要说明的是：对于 ActiveX 控件、插件，浏览器是不会从服务器端下载可执行对象的代码的，而是根据 classid 的定义在本地查找并加载；而 Java Applet 通常是需要根据 URI 从服务器端下载的。

4.2.6　HTML 的变革

1. DHTML 革命

当 Microsoft 和 Netscape 发布其各自浏览器的第 4 版时，Web 开发人员有了一个新的选择：动态 HTML(Dynamic HTML，DHTML)。与有些人想象的不同，DHTML 不是一个 W3C 标准，它更像是一种营销手段。实际上，DHTML 结合了 HTML、层叠样式表(Cascading Style Sheets，CSS)、JavaScript 和 DOM。这些技术的结合使得开发人员可以动态地修改 Web 页面的内容和结构。

注意：

最初 DHTML 的反响很好。尽管 IE 和 Netscape 都支持 DHTML，但各自的实现大相径庭，这要求开发人员必须知道客户使用什么浏览器。而这通常意味着需要大量代码来检查浏览器的类型和版本，这大大增加了开发的开销，因此有些人对于尝试这种方法很是迟疑。

2. XML 及其衍生语言与 Ajax

20 世纪 90 年代中期，基于 SGML 衍生出了 W3C 的可扩展标记语言(eXtensible Markup Language，XML)，自此以后，XML 变得极为流行。XML 将数据和显示方式分开，因而带来了不小的变化。许多人把 XML 视为解决所有计算机开发问题的灵丹妙药，以至于 XML 几乎无处不在。实际上，Microsoft 就已经宣布，Office 12 将支持 XML 文件格式。

Ajax 也是近期比较热门的一项技术。在用户看来，Web 应用相比一般的应用程序构成的应用存在一些问题，其中最突出的问题就是用户界面的交互性不如应用程序的丰富。Web 应用受限于 HTML 所提供的一组基本控件；而更糟糕的是，多年来与服务器交互时必须完全刷新页面。而近期出现的 Google Suggest 和 Gmail 改变了这一现象，这些使用 XMLHttpRequest 对象在浏览器和服务器间完成异步通信的方法改善了用户的体验，为 Web 应用的开发开辟出了一个新的天地。

注意：
本书的第 8 章“Web 展望”将对此进行介绍。

4.3　关于 HTML5

HTML5 草案的前身名为 Web Applications 1.0，于 2004 年被 WHATWG(Web Hypertext Application Technology Working Group，中译为网页超文本技术工作小组)提出，于 2007 年被 W3C 接纳，并成立了新的 HTML 工作团队。 HTML 5 的第一份正式草案已于 2008 年 1 月 22 日公布。HTML5 仍处于完善之中。然而，大部分现代浏览器已经具备了某些 HTML5 支持。

支持的 HTML5 的浏览器包括 Firefox、IE9、Chrome、Safari 等。

4.3.1　HTML5 的特性

1. 语义特性(Class：SEMANTIC)

HTML5 赋予网页更好的意义和结构。更加丰富的标签将随着对 RDFa 的微数据与微格式等方面的支持，构建对程序、对用户都更有价值的数据驱动的 Web。

2. 本地存储特性(Class：OFFLINE & STORAGE)

基于 HTML5 开发的网页 APP 拥有更短的启动时间，更快的联网速度，这些全得益于 HTML5 APP Cache，以及本地存储功能，Indexed DB(HTML5 本地存储最重要的技术之一)和 API 说明文档。

3. 设备兼容特性 (Class：DEVICE ACCESS)

从 Geolocation 功能的 API 文档公开以来，HTML5 为网页应用开发者们提供了更多功能上的优化选择，带来了更多体验功能的优势。HTML5 提供了前所未有的数据与应用接入开放接口。使外部应用可以直接与浏览器内部的数据相连，例如视频影音可直接与 microphones 及摄像头相连。

4. 连接特性(Class：CONNECTIVITY)

更有效的连接工作效率，使得基于页面的实时聊天、更快速的网页游戏体验、更优化的在线交流得到了实现。HTML5 拥有更有效的服务器推送技术，Server-Sent Event 和 WebSockets 就是其中的两个特性，这两个特性能够帮助我们实现服务器将数据“推送”到客户端的功能。

5. 网页多媒体特性(Class：MULTIMEDIA)

支持网页端的 Audio、Video 等多媒体功能，与网站自带的 APPS、摄像头、影音功能相得益彰。

6. 三维、图形及特效特性(Class：3D, Graphics & Effects)

基于 SVG、Canvas、WebGL 及 CSS3 的 3D 功能，用户会惊叹于在浏览器中，所呈现的惊人视觉效果。

7. 性能与集成特性(Class：Performance & Integration)

没有用户会永远等待你的 Loading——HTML5 会通过 XMLHttpRequest 2 等技术，帮助用户的 Web 应用和网站在多样化的环境中更快速地工作。

8. CSS3 特性(Class：CSS3)

在不牺牲性能和语义结构的前提下，CSS3 中提供了更多的风格和更强的效果。此外，较之以前的 Web 排版，Web 的开放字体格式(WOFF)也提供了更高的灵活性和控制性。

HTML5 提供了一些新的元素和属性，例如<nav>(网站导航块)和<footer>。这种标签将有利于搜索引擎的索引整理，同时更好地帮助小屏幕装置和视障人士使用，除此之外，还为其他浏览要素提供了新的功能，如<audio>和<video>标记。 HTML5 的好处包括：

- 使搜索引擎更加容易抓取和索引。对于一些网站，特别是那些严重依赖于 Flash 的网站来说，HTML5 是一大福音。首先，搜索引擎的蜘蛛将能够抓取该站点并索引其中的内容。其次，所有嵌入到动画中的内容将全部可以被搜索引擎读取。在搜索引擎优化的基本理论中，这一方面将会驱动网站获得更多的点击流量。
- 提供更多的功能，提高用户的友好体验。使用 HTML5 的另一个好处就是它可以增加更多的功能，这一点从全球几个主流站点对它的青睐就可以看出。社交网络大亨 Facebook 已经推出他们期待已久的基于 HTML5 的 iPad 应用平台，潘多拉最近也推出他们基于 HTML5 的音乐播放器的新版本，游戏平台 Zynga 最近也推出了三款新的在移动设备浏览器上运行的基于 HTML5 的游戏等。每天都有不断的基于 HTML5 的网站和 HTML5 特性的网站被推出。保持站点处于新技术的前沿，也可以很好地提高用户的友好体验。
- 可用性的提高，提高用户的友好体验。从可用性的角度来看，HTML5 可以更好地促进用户与网站间的互动情况。多媒体网站可以获得更多的改进，特别是在移动平台上的应用，使用 HTML5 可以提供更多高质量的视频和音频流。到目前为止，事实就是 iPhone 和 iPad 将不会支持 Flash，同时 ADOBE 公司也在近期公开声明将停止 Flash 基于移动平台的开发，现在我们已经可以这么说，移动平台日后的多媒体应用将是 HTML5 的天下。

9. HTML5 的改变

HTML5 移除了 HTML4.01 中一系列的标签，主要包括<acronym>、<applet>、<basefont>、<big>、<center>、<dir>、<font>、<frame>、<frameset>、<noframes>、<strike>、<tt>等。

为了更好地处理今天的互联网应用，HTML5 添加了很多新元素及功能，如图形的绘

制、多媒体内容、更好的页面结构、更好的形式处理、API 拖放元素、定位、网页应用程序缓存、存储、网络工作者等，如表 4-8 所示。

表 4-8　HTML5 中新增的标签及其描述

标　签	描　述
<article>	定义页面的侧边栏内容
<audio>	定义音频内容
<bdi>	允许您设置一段文本，使其脱离其父元素的文本方向设置
<command>	定义命令按钮，比如单选按钮、复选框或按钮
<canvas>	标签定义图形，比如图表和其他图像。该标签基于 JavaScript 的绘图 API
<datalist>	定义选项列表。请与 input 元素配合使用该元素，来定义 input 可能的值
<details>	用于描述文档或文档某个部分的细节
<dialog>	定义对话框，比如提示框
<dialog>	定义对话框，比如提示框
<embed>	定义嵌入的内容，比如插件
<figure>	规定独立的流内容(图像、图表、照片、代码等)
<figcaption>	定义<figure>元素的标题
<footer>	定义 section 或 document 的页脚
<header>	定义了文档的头部区域
<keygen>	规定用于表单的密钥对生成器字段
<mark>	定义带有记号的文本
<meter>	定义度量衡，仅用于已知最大和最小值的度量
<nav>	定义运行中的进度(进程)
<progress>	定义任何类型的任务的进度
<output>	定义不同类型的输出，比如脚本的输出
<ruby>	定义 ruby 注释(中文注音或字符)
<rt>	定义字符(中文注音或字符)的解释或发音
<rp>	在 ruby 注释中使用，定义不支持 ruby 元素的浏览器所显示的内容
<section>	定义文档中的节(section、区段)
<source>	定义多媒体资源<video>和<audio>
<track>	为诸如<video>和<audio>元素之类的媒介规定外部文本轨道
<time>	定义日期或时间
<video>	定义视频(video 或者 movie)
<wbr>	规定在文本中的何处适合添加换行符

4.3.2　HTML5 的 canvas

HTML5 为我们带来了很多特性，鉴于其内容较多，此处以比较典型和实用的 canvas

元素为例进行说明。

注意：

此部分内容涉及本书的第 7 章“JavaScript 语言”部分的知识，暂时不能理解的读者可以等学过此内容后再学习。

canvas 提供了通过 JavaScript 绘制图形的方法，此方法使用简单但功能强大。每一个 canvas 元素都有一个“上下文(context)” (可将其想象为绘图板上的一页)，在其中可以绘制任意图形。浏览器支持多个 canvas 上下文，并通过不同的 API 提供图形绘制功能。

1. canvas 基础

创建 canvas 的方法很简单，只需要在 HTML 页面中添加<canvas>元素，程序代码如下：

```
<canvas id="myCanvas" width="300" height="150">
  Fallback content, in case the browser does not support Canvas.
</canvas>
```

为了能在 JavaScript 中引用元素，最好给元素设置 ID；也需要给 canvas 设定高度和宽度。创建好了画布后，就可以通过 JavaScript 在画布中绘制图形了。其一般过程是：首先通过 getElementById 函数找到 canvas 元素，然后初始化上下文；之后可以使用上下文 API 绘制各种图形。【实例 4-16】可实现在 canvas 中绘制矩形、区域擦除并画一些线段。

【实例 4-16】使用 canvas 来绘制图形

下面的例子说明了使用 canvas 进行图形绘制的一般方法，程序代码如 ex4_16.html 所示。

ex4_16.html

```
<!DOCTYPE HTML>
<html>
<body>
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #c3c3c3;">
Your browser does not support the canvas element.
</canvas>
<script type="text/javascript">
// 取得画布元素的引用
var elem = document.getElementById('myCanvas');
 // 总是在使用前检查属性和方法，以确保代码的正确性
if (elem&&elem.getContext) {
  // 获取 2D 的上下文，注意每个元素只有一个上下文环境
  var context = elem.getContext('2d');
  if (context) {
    // 调用 fillRect 函数来画一个矩形，参数分别为(x，y，width，height)
    context.fillStyle="#00FF00";   // 绿色
    context.fillRect(50, 50, 50, 70);   // 填充矩形
    context.strokeStyle = '#FF0000'; // 红色
    context.lineWidth    = 4;   // 线段宽度修改
    context.strokeRect(0,    60, 150, 50);   // 一个空心矩形
```

```
        context.strokeRect(30, 25,   90, 60);
        context.clearRect (30, 25,   90, 60);// 擦除一个区域，形成一个宽度为 2 的细线所画的矩形

        context.moveTo(160,110);   //画线
        context.lineTo(210,10);
        context.lineTo(210,110);
        context.stroke();
    }
}
</script>
</body>
</html>
```

通过 fillStyle 和 strokeStyle 属性可以轻松地设置矩形的填充和线条。颜色值使用方法与在 CSS 中一样，采用十六进制数、rgb()、rgba() 和 hsla()等。

通过 fillRect()可以绘制带填充的矩形；strokeRect()可以绘制只有边框没有填充的矩形；如果想清除部分 canvas，可以使用 clearRect()。这三个方法的参数相同，均为 x、y、width、height。前两个参数设定位置为(x,y)的坐标，后两个参数设置矩形的高度和宽度。lineWidth 属性可以改变线条粗细。

运行后的浏览器显示如图 4-26 所示，在界面上出现了一些图形，请读者对照源码，特别是其中的注释部分，并结合显示效果来理解 canvas 的基本用法。

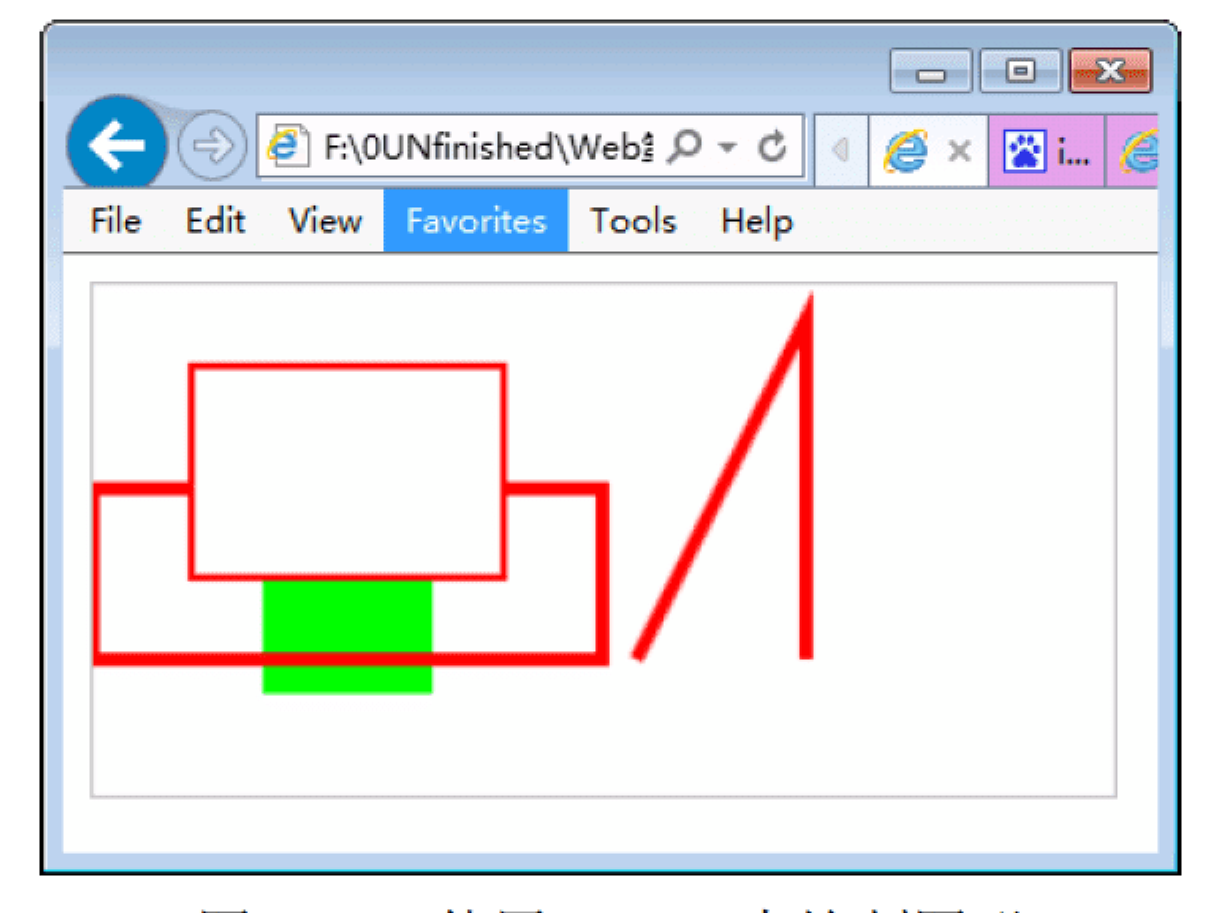


图 4-26　使用 canvas 来绘制图形

2. 路径

通过 canvas 的路径功能(path)可以绘制任意形状。方法是：先绘制轮廓，然后绘制边框和填充。创建自定义形状很简单，只需要先使用 beginPath()开始绘制，然后使用直线、曲线和其他图形进一步完成绘制。绘制完毕后调用 fill 和 stroke 即可添加填充或者设置边框，调用 closePath()则结束自定义图形的绘制。

【实例 4-17】使用 canvas 的 path 功能来绘制图形

下面的例子说明了使用 canvas 的 path 功能来进行图形绘制的方法，程序代码如 ex4_17.html 所示。

ex4_17.html

```
<!DOCTYPE HTML>
<html>
<body>
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #c3c3c3;">
Your browser does not support the canvas element.
</canvas>
```

```
<script type="text/javascript">

var elem = document.getElementById('myCanvas');
if (elem&&elem.getContext) {
    var context = elem.getContext('2d');
    if (context) {

        context.fillStyle     = '#0000ff';
        context.strokeStyle = '#ff0000';
        context.lineWidth     = 4;

        context.beginPath();
        context.moveTo(10, 10); //  设置三角形的坐标
        context.lineTo(100, 10);
        context.lineTo(10, 100);
        context.lineTo(10, 10);

//  调用下面的方法后可以使形状显示出来
        context.fill();    //  若将词句代码放在 closePath 之后，则红色的外框变窄一半
        context.stroke();   //  三角形的外框
        context.closePath();

        context.fillStyle="#00FF00";    //  绿色
        context.beginPath();
        context.arc(245,100,30,80,Math.PI*2,true);    //  画圆
        context.closePath();
        context.fill();
    }
}
</script>
</body>
</html>
```

运行后的浏览器显示如图 4-27 所示，在界面上出现了一个带边框的蓝色三角形和一个圆形，请读者对照源码，特别是其中的注释部分，并结合显示效果来理解 canvas 中利用 path 来绘图的用法。

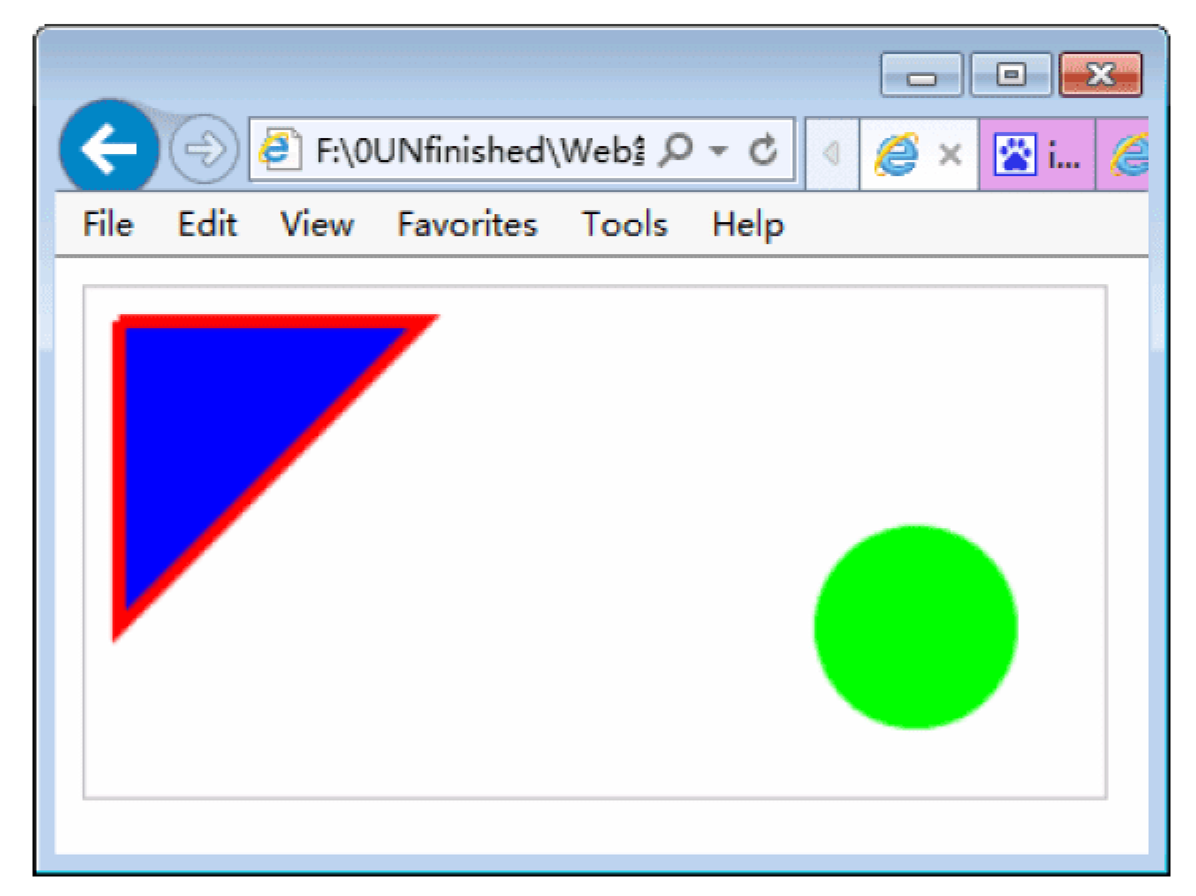


图 4-27　使用 canvas 的 path 功能来绘制图形

3. 颜色渐变效果的生成

用 fillStyle 和 strokeStyle 属性可以设置成 CanvasGradient 对象，即用 CanvasGradient 为线条和填充使用颜色渐变。

如果希望创建一个 CanvasGradient 对象，可以使用以下两个方法：createLinearGradient() 和 createRadialGradient()，其中前者创建线性颜色渐变，后者创建圆形颜色渐变。创建颜色渐变对象后，则可以使用对象的 addColorStop 方法添加颜色中间值，其用法如下。

```
CanvasGradient = ctx.createLinearGradient(x0, y0, x1, y1)
```

```
CanvasGradient.addColorStop(offset, color)
```

createLinearGradient()方法中设置了渐变的开始坐标(x0, y0)和结束坐标(x1, y1)，函数会返回线性渐变对象 CanvasGradient。 然后通过 addColorStop()方法，offset 为 0 的地方为开始地点的颜色，offset 为 1 的地方为结束地点的颜色。另外，很明显的，x0=x1 并且 y0=y1 的时候，不会有渐变效果出现。如果 offset 是指定 0~1 之间的值，则是对应中间的比例位置。

【实例 4-18】使用 canvas 的颜色渐变功能

下面的例子说明了使用 canvas 的颜色渐变功能来进行图形填充的方法，程序代码如 ex4_18.html 所示。

ex4_18.html

```
<!DOCTYPE HTML>
<html>
<body>
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #c3c3c3;">
Your browser does not support the canvas element.
</canvas>
<script type="text/javascript">

var elem = document.getElementById('myCanvas');
if (elem&&elem.getContext) {
   var context = elem.getContext('2d');
   if (context) {
      var grd=context.createLinearGradient(0,0,150,100);
      grd.addColorStop(0,"#0000FF");
      grd.addColorStop(1,"#00FFFF");
      context.fillStyle=grd;
      context.fillRect(0,0,300,150);    //填充
   }
}
</script>
</body>
</html>
```

运行后的浏览器显示如图 4-28 所示，在界面上出现了一个利用渐变色进行填充的矩形，请读者对照源码，并结合显示效果来理解 canvas 中渐变色的用法。

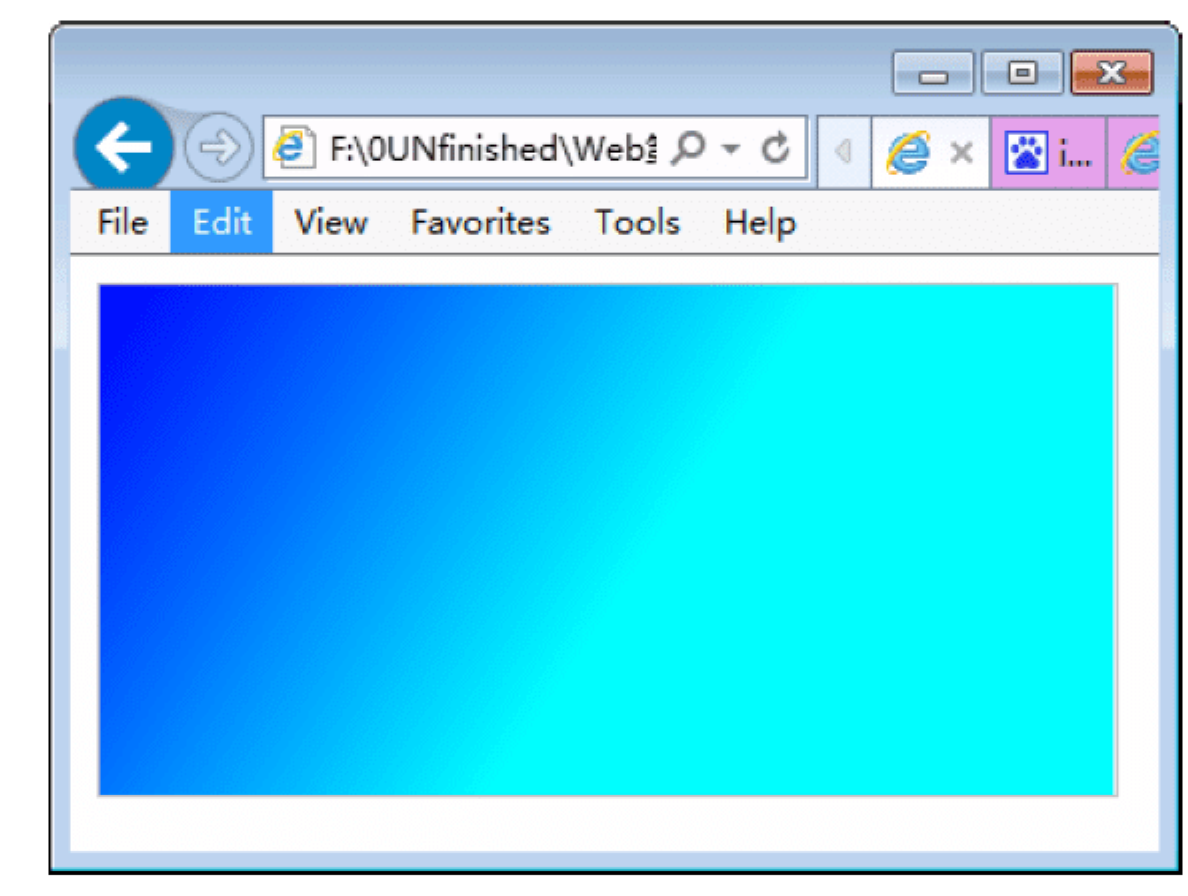


图 4-28　使用 canvas 的渐变色填充

4.3.3　关于<!DOCTYPE>声明

<!DOCTYPE>声明位于文档中的最前面的位置，处于<html>标签之前。此标签可通知浏览器文档使用了什么 HTML 或 XHTML 规范。因此，为了获得正确的 DOCTYPE 声明，

关键就是让 DTD 与文档所遵循的标准对应。

在 DOCTYPE 标签中可声明三种 DTD 类型，分别表示严格版本、过渡版本以及基于框架的 HTML 版本。如果文档中的标记不遵循 DOCTYPE 声明所指定的 DTD，这个文档除了不能通过代码校验之外，还有可能无法在浏览器中正确显示。

另外，如果 DOCTYPE 声明指定的是 XHTML DTD，但文档包含的是旧式风格的 HTML 标记，就是不恰当的；类似地，如果 DOCTYPE 声明指定的是 HTML DTD，但文档包含的是 XHTML1.0 strict 标记，同样是不恰当的。

<!DOCTYPE>的典型用法为：

```
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

在上面的声明中，声明了文档的根元素是 html，它在公共标识符被"-//W3C//DTD XHTML 1.0 Strict//EN" 的 DTD 中进行了定义。浏览器可以据此寻找到匹配此公共标识符的 DTD。如果找不到，浏览器将使用公共标识符后面的 URL 寻找 DTD。对此定义中的各部分说明如下。

- -：表示组织名称未注册。Internet 工程任务组(IETF)和万维网协会(W3C)并非注册的 ISO 组织。
- +：为默认，表示组织名称已注册。
- DTD：指定公开文本类，即所引用的对象类型，默认为 DTD。
- HTML：指定公开文本描述，即对所引用的公开文本的唯一描述性名称，后面可附带版本号，默认为 HTML。
- URL：指定所引用对象的位置。
- Strict：排除所有 W3C 专家希望逐步淘汰的代表性属性和元素。

如果没有指定有效的 DOCTYPE 声明，大多数浏览器都会使用一个内建的默认 DTD 。在这种情况下， 浏览器会用内建的 DTD 来试着显示所指定的标记。

1. HTML 4 的文档类型

在 HTML 4.01 中规定了三种文档类型：Strict、Transitional 以及 Frameset。

- 如果需要干净的标记，免于表现层的混乱，用 HTML Strict DTD 类型，用法如下。

```
    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "
http://www.w3.org/TR/html4/strict.dtd">
```

- Transitional DTD 中可包含 W3C 所期望移入样式表的呈现属性和元素。如果用户使用了不支持层叠样式表(CSS)的浏览器以至于不得不使用 HTML 的呈现特性时，可以用 Transitional DTD 类型，用法如下。

```
    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "
http://www.w3.org/TR/html4/loose.dtd">
```

- Frameset DTD 被用于带有框架的文档。除 frameset 元素取代了 body 元素之外，

Frameset DTD 等同于 Transitional DTD，用法如下。

```
    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "
http://www.w3.org/TR/html4/frameset.dtd">
```

2. XML 的文档类型

XHTML 1.0 规定了三种 XML 文档类型：Strict、Transitional 以及 Frameset。

- 如果需要干净的标记，免于表现层的混乱，用 XHTML Strict DTD 类型，用法如下。

```
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

- Transitional DTD 可包含 W3C 所期望移入样式表的呈现属性和元素。如果用户使用了不支持层叠样式表(CSS)的浏览器以至于不得不使用 HTML 的呈现特性时，用 Transitional DTD 类型，用法如下。

```
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

- Frameset DTD 被用于带有框架的文档。除 frameset 元素取代了 body 元素之外，Frameset DTD 等同于 Transitional DTD，用法如下。

```
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

3. 选取方法

互联网上很多网站(如百度、谷歌等知名网站)直接使用了“<!DOCTYPE html>”，很多 HTML5 网站也直接用的这个方式，这是因为 HTML 5 不是基于 SGML 的，因此不需要对 DTD 进行引用，但是需要 DOCTYPE 来规范浏览器的行为，也就是让浏览器按照它们应有的方式来运行。

注意：

浏览器对于各种格式的支持是动态变化的，也许读者看到本书的时候有些浏览器已经能支持相应的格式了，也可能已经出现了新的浏览器。只要读者自行对相应格式和浏览器进行测试，就能准确获知其支持的情况。建议采用“<!DOCTYPE html>”方式，因为此方式可以开启浏览器的标准兼容模式。在标准兼容模式下，虽然不能保证与其他版本(指的是 IE 6 之前的)的 Internet Explorer 或其他浏览器保持兼容，文档的渲染行为也许与将来的浏览器不同，但不失为一种既简单又实用的声明方式。

为了兼容，本书的部分例子采用了“<!DOCTYPE html>”方式，也有部分例子做了 DTD 声明，读者可以试着作些修改并查看不同浏览器中的显示效果以进行验证。

4.3.4　一个 HTML5 实例——Web 上的视频

今天，大多数视频是通过插件(比如 Flash)来显示的。然而，并非所有浏览器都拥有同样的插件。HTML5 规定了一种通过 video 元素来包含视频的标准方法。

当前，video 元素支持 3 种视频格式，分别如下。

- Ogg：带有 Theora 视频编码和 Vorbis 音频编码的 Ogg 文件。
- MPEG4：带有 H.264 视频编码和 AAC 音频编码的 MPEG 4 文件。
- WebM：带有 VP8 视频编码和 Vorbis 音频编码的 WebM 文件。

支持<video>元素的浏览器及版本说明如表 4-9 所示。

表 4-9　支持<video>元素的浏览器及版本说明

格　式	IE	Firefox	Opera	Chrome	Safari
Ogg	不支持	3.5+	10.5+	5.0+	不支持
MPEG 4	9.0+	不支持	不支持	5.0+	3.0+
WebM	不支持	4.0+	10.6+	6.0+	不支持

【实例 4-19】使用 HTML5 的 video 元素

下面的例子说明了 HTML5 的 video 元素的用法，程序代码如 ex4_19.html 所示。

ex4_19.html

```
<video src="dean.mp4" width="320" height="240" controls="controls">
Your browser does not support the video tag.
</video>
```

上面的例子使用了 MP4 格式的文件，当然对于 Firefox、Opera 以及 Chrome 等多种浏览器，视频的格式有很多种选择，而如果希望也能适用于 Safari 浏览器，则视频文件最好是 MPEG 4 格式。该实例在支持 HTML5 的浏览器中运行时就可以看到视频被打开和播放，其中在视频显示区域下部有一个交互控制和信息显示半透明条，可用于控制和显示，如图 4-29 所示。

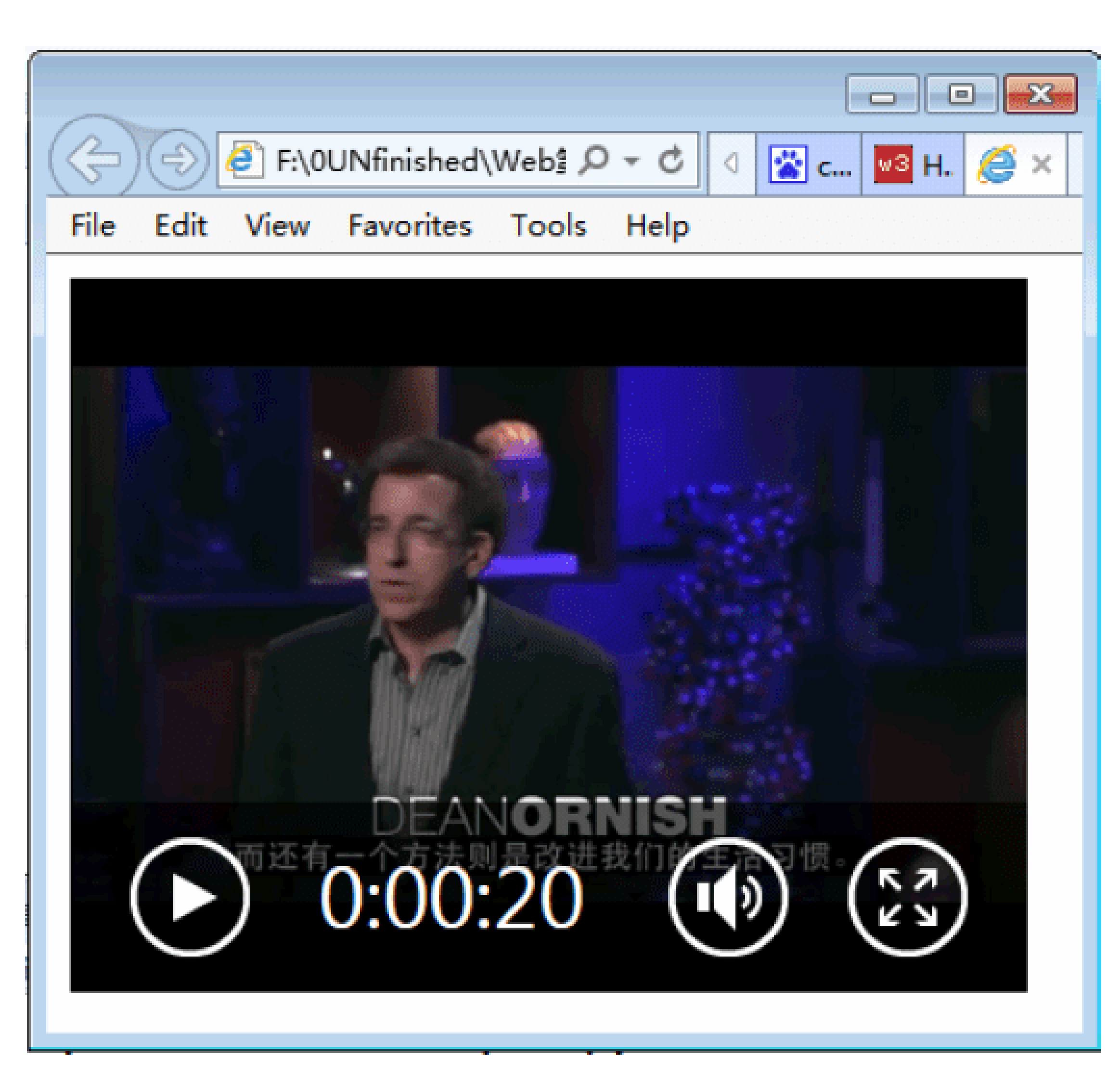


图 4-29　video 标签的使用

video 标签支持的属性如表 4-10 所示。

表 4-10　video 标签的属性

属　性	值	描　述
autoplay	autoplay	如果出现该属性，则视频在就绪后马上播放
controls	controls	如果出现该属性，则向用户显示控件，比如播放按钮
height	pixels	设置视频播放器的高度
loop	loop	如果出现该属性，则当媒介文件完成播放后再次开始播放
preload	preload	如果出现该属性，则视频在页面加载时进行加载，并预备播放；如果使用 "autoplay"，则忽略该属性
src	url	要播放的视频的 URL
width	pixels	设置视频播放器的宽度

4.4　本章小结

本章讲解了交互设计、HTML 的高级应用以及 HTML5 的相关知识。重点介绍了人机交互和框架、表单、脚本、多媒体及 HTML5 等方面的应用。

4.5　思考和练习

1. 使用编写工具生成的 HTML 源代码是缩进的，但某些网页的代码却没有设置缩进，缩进有什么作用？

2. 一些网页在 IE 和 Navigator 中都能够非常好地工作，但是在其他浏览器(如 360 等)中显示有一些问题。怎样才能在不放弃网页中那些出现问题的部分的情况下做到兼容？

3. 如果一个视频剪辑或者其他文件已经存在另一个网站中，能否仅仅链接到它而不用在本网站中提供此文件的副本？

4. 使用标签时，有时是以大写输入的(<TITLE>)而另外的时候又以小写(<title>)出现。它们有区别吗？

5. 对于之前编写的 HTML4 的网站，如何快速地变迁到 HTML5？

第5章　层叠样式表(CSS)

W3C(World Wide Web Consortium)把动态 HTML(Dynamic HTML)分为三个部分来实现：支持动态效果的浏览器(包括 Internet Explorer、Netscape Navigator 等)、CSS 样式表和脚本语言(包括 JavaScript、Vbscript 等)。

CSS 可以为网页设计者的网页空间应用提供更大的弹性，让网页的文字内容与版面设计分开处理。因此，CSS 在网站设计过程中得到了广泛的应用。本章旨在引导读者了解 CSS(Cascading Stylesheets，层叠样式表)，它是制作网站过程中一项重要的技术，目前大多数浏览器对此提供了支持。本章除了介绍 CSS 的基本概念、各组成部分、滤镜的使用以外，还说明了在 Dreamweaver 中运用 CSS 的方法，并通过若干实例介绍了 CSS 的典型用法。

CSS3 是 CSS 技术目前最新的版本，CSS3 是向着模块化发展的。以前的规范作为一个模块实在是太庞大而且比较复杂，所以，把它分解为一些小的模块，更多新的模块也被加入进来。这些模块包括： 盒子模型、列表模块、超链接方式、语言模块、背景和边框、文字特效、多栏布局等。

本章要点：

- 理解 CSS 与网页制作之间的关系
- 在网页中使用 CSS 的基本方法
- CSS 滤镜的用法
- CSS3 及其基本用法
- 网页中应用 CSS 的典型案例

5.1　CSS 概述

CSS(Cascading Stylesheets，层叠样式表)可以为网页设计者的网页空间应用提供更大的弹性，让网页的文字内容与版面设计分开处理。因此，CSS 在网站设计过程中得到了广泛的应用，也是制作网站过程中一项重要的技术。

本书第 3 章介绍了 HTML 的有关知识，最初的网页设计就是用 HTML 标签来定义页面文档及其格式的，例如标题<H1>、段落<P>、表格<TABLE>、链接<A>等。但这些标签中所规定的样式是独立设置的，由于不能完全满足复杂文档在样式管理方面的需求，W3C(The World Wide Web Consortium)于 1997 年在颁布 HTML4 标准的同时也公布了有关样式表的第一个标准 CSS1，1998 年 5 月又发布了 CSS2 版本，以及最近发布了 CSS3，至此样式表得到了更多的充实。使用 CSS 可以精确地控制页面里每一个元素的字体样式、背

景、排列方式、区域尺寸、四周加入边框等；使用 CSS 还能够简化网页的格式代码，加快下载及显示的速度，外部链接样式可以同时定义多个页面，大大减少了重复劳动的工作量。

注意：

CSS 需要浏览器的支持才能发挥作用，如果所使用的浏览器不支持或者某些浏览器的版本较低可能会造成效果上的损失。

CSS 为网页带来的好处如下：

- 目前几乎所有的浏览器都支持。
- 一部分通过图像处理软件来实现特效功能，通过 CSS 的滤镜也可以实现。这不仅能加快页面的装载速度，也能借助这一点实现网页的动态效果。
- 页面的字体等样式变得更易于管理，方便编排，便于网站风格和样式的调整和修改，简化界面样式管理成本。
- 可以将站点上所有的网页风格都使用一个外部 CSS 文件进行控制，只要修改这个 CSS 文件中相应的行，那么整个站点的所有页面都会随之发生变动，实现更新网站时让众多网页的风格样式同步更新，不用一页一页更新，不仅减少了大量重复性的工作，而且有利于以后的修改、编辑，浏览时也减少了重复下载代码的次数。

对于小型、简单的网站，如个人网站，由于规模较小也可以不使用 CSS，使用 CSS 某种程度上可能反而会增加开发的成本和难度，还需要付出额外时间来学习。但对于中型以上的网站，使用 CSS 无疑带来好处，因此 Internet 上不使用 CSS 的网站非常少。

5.2　为网页添加样式表的方法

要想使用 CSS 来进行样式的设定和管理，首先就要让浏览器识别并调用。当浏览器读取网页时，对于其中的样式表也按照文本格式来读取，因此需要一种明确的标识以便浏览器识别样式表的描述。一般而言，有三种在页面中插入样式表的方法：内嵌样式、内部样式表、外部样式表。

1. 内嵌样式

内嵌样式是混合在 HTML 标记里使用的，用这种方法，可以仅对某个元素单独定义样式。内嵌样式的使用是直接在 HTML 标签中加入 style 参数。而 style 参数的内容就是 CSS 的属性和值，定义方式如下：

<标签 style="参数 1:值 1; 参数 2: 值 2">网页的内容< /标签>

例如：

```
<P style="color: blue; font-size: 10pt">CSS 样式表</P>
```

上面定义了用蓝色显示字体大小为 10pt 的“样式表”。尽管使用简单、显示直观，但

这种方法不常用，因为它无法完全发挥样式表对样式进行统一管理的优势。与其这样使用样式表，还不如直接利用标签和属性进行定义。

style 参数可以应用于<BODY>内任意的元素(包括<BODY>本身)，除了 BASEFONT、PARAM 和 SCRIPT。

2. 内部样式表

统一在 HTML 的头部标签<HEAD>中添加，例如：

```
<HEAD>
  <STYLE type="text/css">
      <!--
        P{color:blue;font-size:10pt}
      -->
  </STYLE>
</HEAD>
```

为了帮助不支持 CSS 的浏览器过滤掉 CSS 代码，避免在浏览器中直接以源代码的方式显示这里设置的样式表，有必要在样式表里加上注释标识符“<!--注释内容-->”，以达到隐藏这些内容而不让它显示出来的目的。

3. 外部样式表

外部样式表是一个单独保存的文件，其中定义了一些样式可供调用。由于以独立文件的方式保存，因此可以供所有网页来使用，起到统一控制的作用。将外部样式表引入网页的方式有两个：使用<LINK>标签或用@import 语句。

样式表文件可以用任何文本编辑器(如记事本)打开并编辑，一般样式表文件扩展名必须为.css。内容是定义的样式表，其中不包含<HTML>标签，假设 aDefinedCSS.css 文件的内容如下：

```
hr {color: blue}
p {text-align: center;}
body {background-image: url("images/background.jpg")}
```

该说明定义了：水平线的颜色为蓝色；段落为居中对齐；页面的背景图片为 images 目录下的 background.jpg 文件。

在页面中使用<LINK>标签可以链接到某个样式表文件，该<LINK>标签是添加在 HTML 的头部标签<HEAD>里的，如果需要引入上面所定义的样式表文件“aDefinedCSS.css”，则可以通过下面的方法：

```
<HEAD>
  <LINK rel="stylesheet" href="aDefinedCSS.css" type="text/css">
</HEAD>
```

上面的语句表示浏览器从 aDefinedCSS.css 文件中以文档格式读出已定义好的样式表，aDefinedCSS.css 是需要引入的样式表文件。需要注意的是这个样式表文件中不能包含

<STYLE>标签，并且只能以 css 作为后缀名。rel="stylesheet"表示样式表将读取一个外部文件形式的样式表再与 HTML 文档结合。type="text/css"是指文件的类型是样式表文本。href="aDefinedCSS.css"是文件所在的位置。

对于同样的“aDefinedCSS.css”文件，也可以利用@import 语句来引入。这时需要在内部样式表的<style>里使用@import 语句，这一点与 Link 标签是完全不同的，例如：

```
<HEAD>
   <STYLE type="text/css">
      < !--
      @import " aDefinedCSS.css"
      -->
   </STYLE>
</HEAD>
```

上例中“@import "aDefinedCSS.css"”表示导入 aDefinedCSS.css 样式表，注意使用时外部样式表的路径必须正确。正确导入后其中定义的样式即可生效。

注意：

使用@import 方式导入外部样式表必须在样式表的开始部分，在其他内部样式表之前。

4. 优先级

如果在同一处使用了多个不同的样式时，将会产生这几个样式表相叠加的效果，当然其中有时就会产生冲突，此时需要明确一般会以最靠近的样式表为准。例如：首先链入一个外部样式表，其中定义了 h3 选择符的 color、text-align 和 font-size 属性：

```
h3
{
   color: red;
   text-align: left;
   font-size: 8pt
}
```

这里定义的 h3 的文字颜色为红色；向左对齐；文字尺寸为 8 号字。然后在内部样式表里同样定义了 h3 选择符的 text-align 和 font-size 属性：

```
h3
{
   text-align: right;
   font-size: 20pt
}
```

这次定义 h3 为文字向右对齐；文字尺寸为 20 号字。那么这个页面叠加后的样式就是：

```
color: red;
text-align: right;
font-size: 20pt
```

即：文字颜色为红色；向右对齐；文字尺寸为 20 号字。这里内部样式表中未定义颜色，因此字体颜色从外部样式表里保留下来，而对齐方式和字体尺寸这两项都有定义时，按照后定义的优先，因此最终依照内部样式表来显示。

注意：

依照后定义优先的原则，优先级从高到低排列如下：内嵌样式、内部样式表、导入外部样式表。而外部样式表和内部样式表之间遵循后定义的样式优先级更高的原则。

5.3　用 CSS 定义样式

5.3.1　简单的 CSS 应用

CSS 的定义由三个部分构成：选择符(selector)、属性(properties)和属性的取值(value)，基本格式如下：

```
selector {property: value}
```

其实就是“选择符 {属性：值}”的格式。其中的选择符是可以有多种形式，一般是所要定义样式的 HTML 标签，如<BODY>、<P>、<TABLE>等，通过上面的方法可以定义其属性和值，但属性和值之间需要用冒号隔开，如：

```
body {color: black}
```

选择符 body 是指页面主体部分，color 是控制文字颜色的属性，black 是颜色的值，此例的效果是使页面中的文字显示为黑色。

如果属性的值是多个单词组成，必须在值上加引号，如字体的名称经常是几个单词的组合，下面的定义将段落字体说明为 sans serif：

```
p {font-family: "sans serif"}
```

如果需要对一个选择符指定多个属性时，可以用分号将所有的属性和值分开，如下所示：

```
p {text-align: center; color: red}
```

这个定义将段落居中排列；并且段落中的文字为红色。

为了使所定义的样式表方便阅读，也可以采用分行的书写格式：

```
p
{
   text-align: center;
   color: black;
   font-family: arial
}
```

此处定义了：段落排列居中，段落中文字为黑色，字体是 arial。

【实例 5-1】一个简单的 CSS 应用

程序代码如 ex5_1.html 所示。

ex5_1.html

```
<HTML>
  <HEAD>
    <STYLE>
      P
      {
        text-align: center;
        color: black;
        font-family: arial
      }
    </STYLE>
  </HEAD>
  <BODY>
    <H3 align="right" color="blue">
      利用 HTML 标签很复杂
    </H1>
    <P>利用  CSS  更简单</P>
  </BODY>
</HTML>
```

在这个例子中选择符为 P，下面定义了 3 个属性：text-align、color 和 font-family，它们的取值分别为 center、black 和 arial。

该实例运行后的浏览器显示如图 5-1 所示，图中的上面一行文字使用了一般标签的属性来控制显示样式；而下面一行文字使用了在<HEAD>部分<STYLE>标签中定义的样式来显示。



图 5-1　一个简单的 CSS 应用

5.3.2　选择符组

为了减少样式的重复定义，可将相同属性和值的选择符组合起来书写，使用逗号将选择符分开，例如：

```
h1, h2, h3, h4, h5, h6 { color: green }
```

该定义包括了这个组里所有的标题元素——(h1, h2, h3, h4, h5, h6)，将这个组里每个标

题元素的文字设置为绿色。

又如：

```
p, table{ font-size: 9pt }
```

该定义将段落和表格里的文字尺寸设置为 9 号字，其效果完全等效于：

```
p { font-size: 9pt }
table { font-size: 9pt }
```

【实例 5-2】使用选择符组

程序代码如 ex5_2.html 所示。

ex5_2.html

```
<HTML>
  <HEAD>
    <STYLE>
      h1, h2, h3, h4, h5, h6 {color:green}
    </STYLE>
  </HEAD>
  <BODY>
    <H1>直接使用 H1 标签</H1>
    <H3>直接使用 H3 标签</H3>
    <H2 align=right>使用 H2 标签并加上右对齐</H2>
  </BODY>
</HTML>
```

在这个例子中的选择符组对于 H1 至 H6 同时做了显示为绿色的声明，在<BODY>中，直接使用了<H1>和<H3>，但对于<H2>增加了 align 的设置。

该实例运行后的浏览器显示如图 5-2 所示，所有文字均显示为绿色，其中 H2 的一行显示为右对齐。

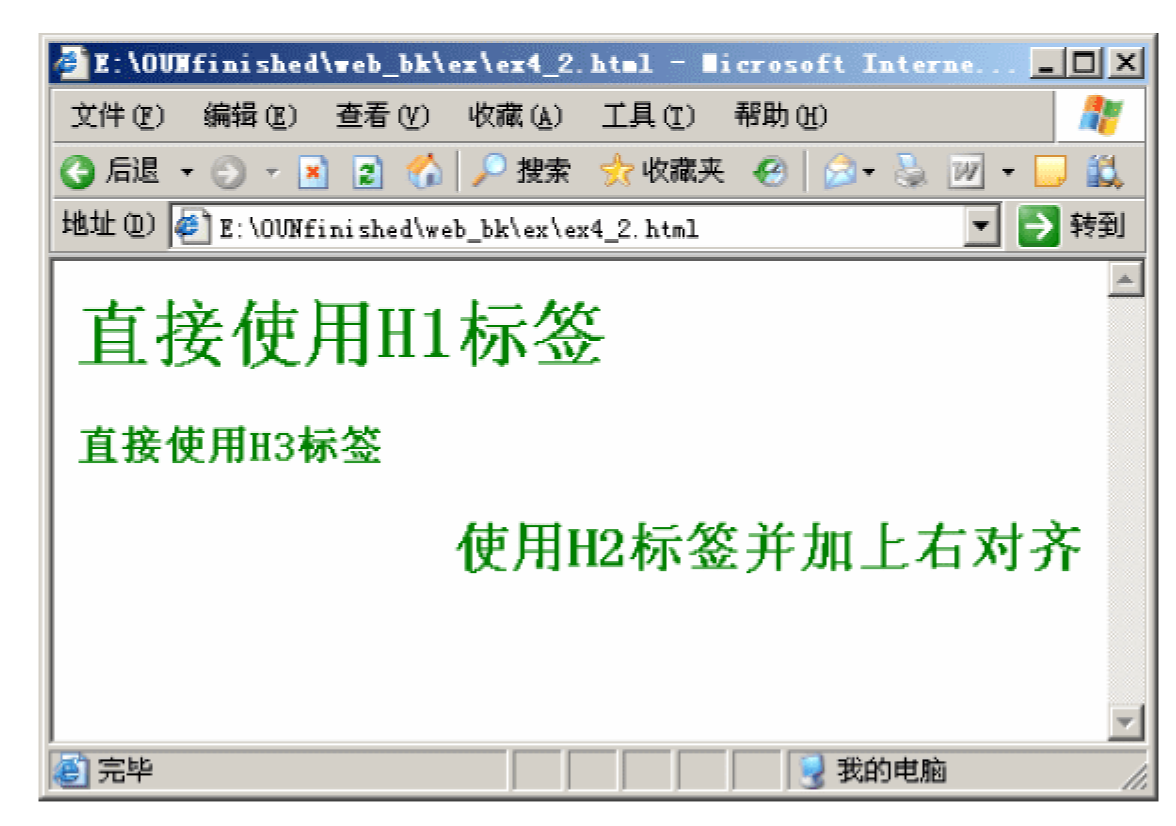


图 5-2 选择符组的使用

5.3.3 类选择符

用类选择符可以将相同的元素分类定义为不同的样式，定义类选择符时，在自定义类的名称前面加一个点号。若要定义两个不同的段落，一个段落向右对齐，一个段落居中，则可先定义两个类——p.right 和 p.center：

```
P.right {text-align: right}
P.center {text-align: center}
```

然后将它们分别用在不同的段落里，此时只要在 HTML 标记里将刚才定义的类以 class 参数的形式加入相应的元素中即可，如下所示：

```
<P class="right">
  这个段落是向右对齐的
</P>
<P class="center">
  这个段落是居中排列的
</P>
```

类的名称可以是任意英文单词或以英文开头与数字的组合，一般而言，为了帮助记忆和方便调用，常以其功能和效果的名称简写后的形式来命名。

【实例 5-3】使用类选择符

程序代码如 ex5_3.html 所示。

ex5_3.html

```
<HTML>
  <HEAD>
    <STYLE>
      p.right {text-align: right}
      p.center {text-align: center}
    </STYLE>
  </HEAD>
  <BODY>
    <p class="right">
      这个段落是向右对齐的
    </p>
    <p class="center">
      这个段落是居中排列的
    </p>
  </BODY>
</HTML>
```

该例子对所定义的 CSS 样式分别给出了不同的名称，运行后的浏览器显示如图 5-3 所示，最终的效果达到了设计的目的。

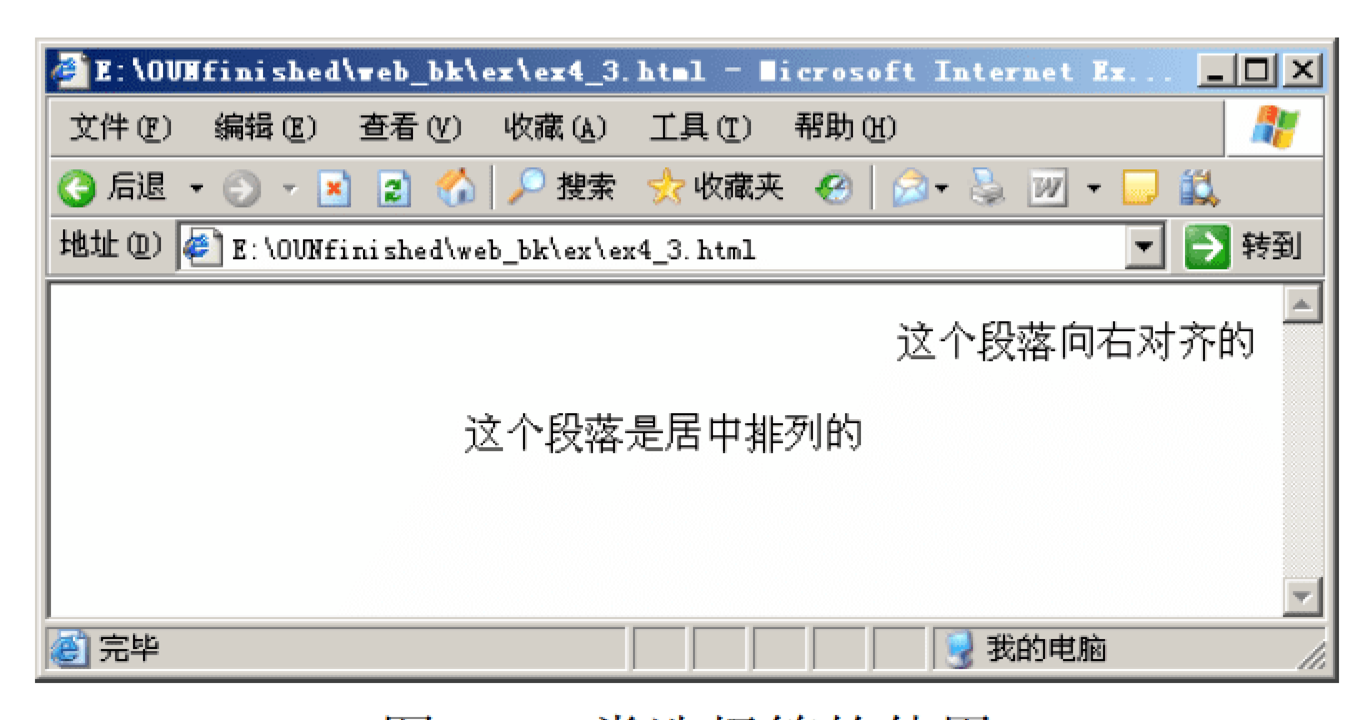


图 5-3　类选择符的使用

类选择符还有一种用法，如果在定义选择符时省略 HTML 标签名，就可以把几个不同的元素定义成相同的样式：

```
.center {text-align: center}
```

这个定义将.center 的类选择符设为文字居中排列。这样的类可以被应用到任何元素上。

下面我们使 H1 元素(标题 1)和 P 元素(段落)都归为“center”类，这使得两个不同元素的样式都跟随“.center”这个类选择符：

```
<H1 class="center">
   这个标题 H1 是居中排列的
</H1>
<P class="center">
   这个段落 P 也是居中排列的
</P>
```

注意：

这种省略 HTML 标记的类选择符是一种最常用的 CSS 方法，使用这种方法，可以很方便地在任意元素上套用预先定义好的类样式。也就是说成为一种通用的类样式，不受标签名称的限制。

【实例 5-4】使用通用类选择符

程序代码如 ex5_4.html 所示。

ex5_4.html

```
<HTML>
   <HEAD>
      <STYLE>
         .center {text-align: center}
      </STYLE>
   </HEAD>
   <BODY>
      <H1 class="center">
         这个标题 H1 是居中排列的
      </H1>
      <P class="center">
         这个段落 P 也是居中排列的
      </P>
   </BODY>
</HTML>
```

在该例中，所定义的通用类选择符 center，无论是对于<H1>还是<P>，只要增加了属性 class 并设置为前面所定义的值，就可以使用前面的设定，运行后的浏览器显示如图 5-4 所示。

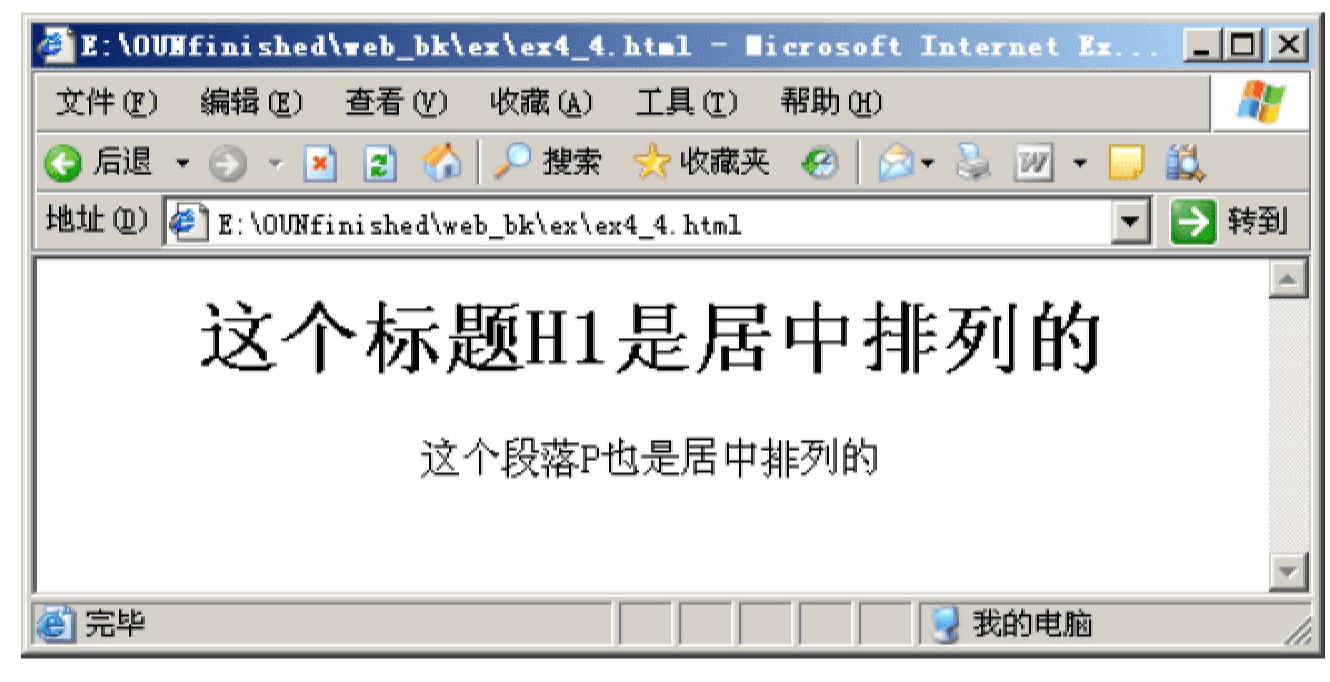


图 5-4　通用类选择符的使用

5.3.4　ID 选择符

在 HTML 页面中 ID 参数指定了单一的元素，ID 选择符用来对这个元素指定单独的样式。ID 选择符的应用和类选择符类似，只要把 class 换成 ID 即可，程序代码如下：

```
<P id="intro">
  这个段落向右对齐
</P>
```

定义 ID 选择符时需要在 ID 名称前加上一个“#”号。和类选择符相同，定义 ID 选择符的属性也有两种方法。一种方法不限制匹配的元素；另一种方法对进行匹配的元素进行了限制。

【实例 5-5】使用不限制匹配的 ID 选择符

程序代码如 ex5_5.html 所示。

ex5_5.html

```
<HTML>
  <HEAD>
    <STYLE>
      #intro
      {
        font-size:150%;
        font-weight:bold;
        color:#ff0000;
      }
    </STYLE>
  </HEAD>
  <BODY>
    <H1 id="intro">
      这个标题 H1 是红色加粗，按 150%的比例显示的
    </H1>
    <P id="intro">这个段落是红色加粗，按 150%的比例显示的</P>
  </BODY>
</HTML>
```

本实例中，ID 属性将匹配所有 id="intro"的元素，无论对于<H1>还是<P>，只要增加了属性 id 并设置为“intro”，就可以直接应用所定义的样式。此处的样式设置为：字体尺寸为默认尺寸的 150%，粗体，红色。该实例运行后的浏览器显示如图 5-5 所示。可见，当样式中进行了较多的设置时，使用本例中的这种方式，可以简化 CSS 样式的定义工作量。

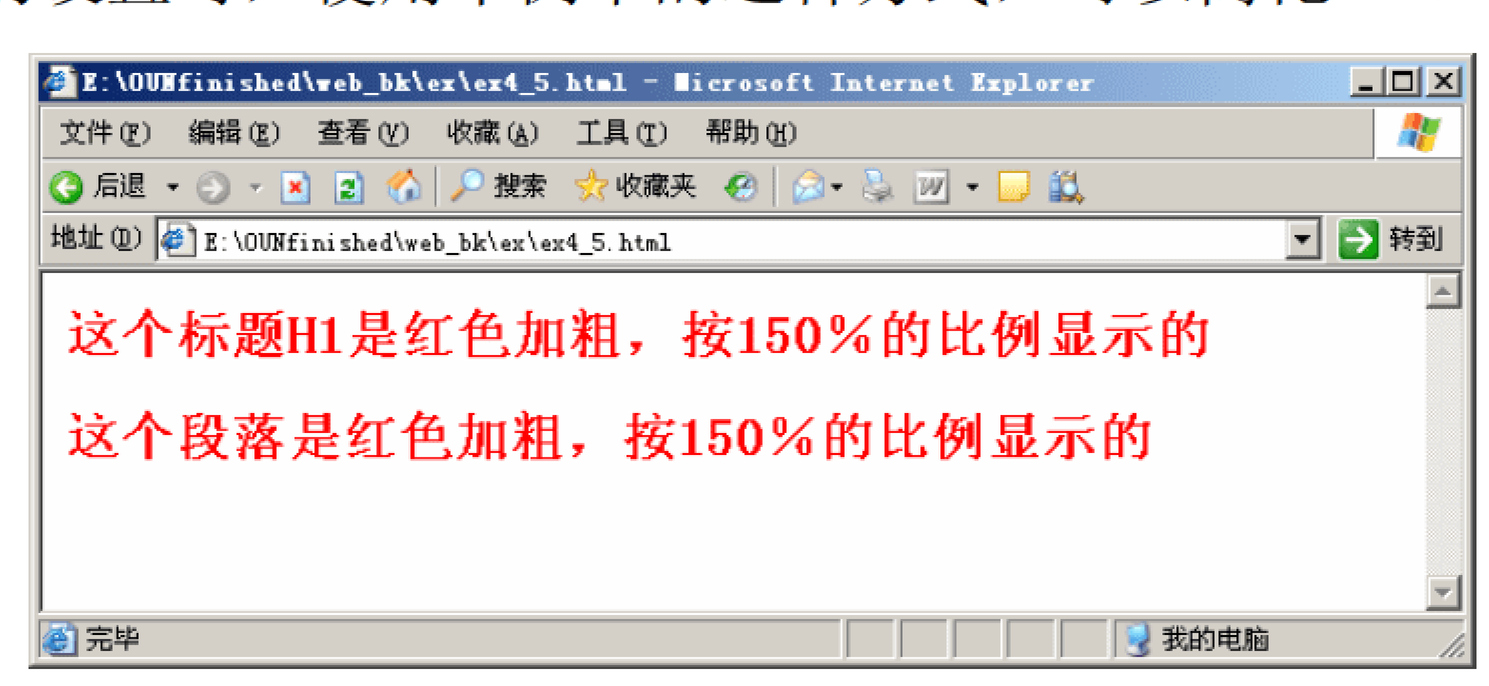


图 5-5　使用不限制匹配的 ID 选择符

【实例 5-6】使用限制匹配的 ID 选择符

程序代码如 ex5_6.html 所示。

ex5_6.html

```
<HTML>
  <HEAD>
    <STYLE>
      P#intro
      {
        font-size:150%;
        font-weight:bold;
        color:#ff0000;
      }
    </STYLE>
  </HEAD>
  <BODY>
    <H1 id="intro">
      这个标题 H1 是直接显示的
    </H1>
    <P id="intro">这个段落是红色加粗，按 150%的比例显示的</P>
  </BODY>
</HTML>
```

本实例中，ID 属性只匹配 id="intro"的段落元素，因此对于<H1>，增加的 id="intro"属性是无效的，但对于段落元素<P>，却可以使用。此处定义的样式为：字体尺寸为默认的 150%，粗体，红色。该实例运行后的浏览器显示如图 5-6 所示。

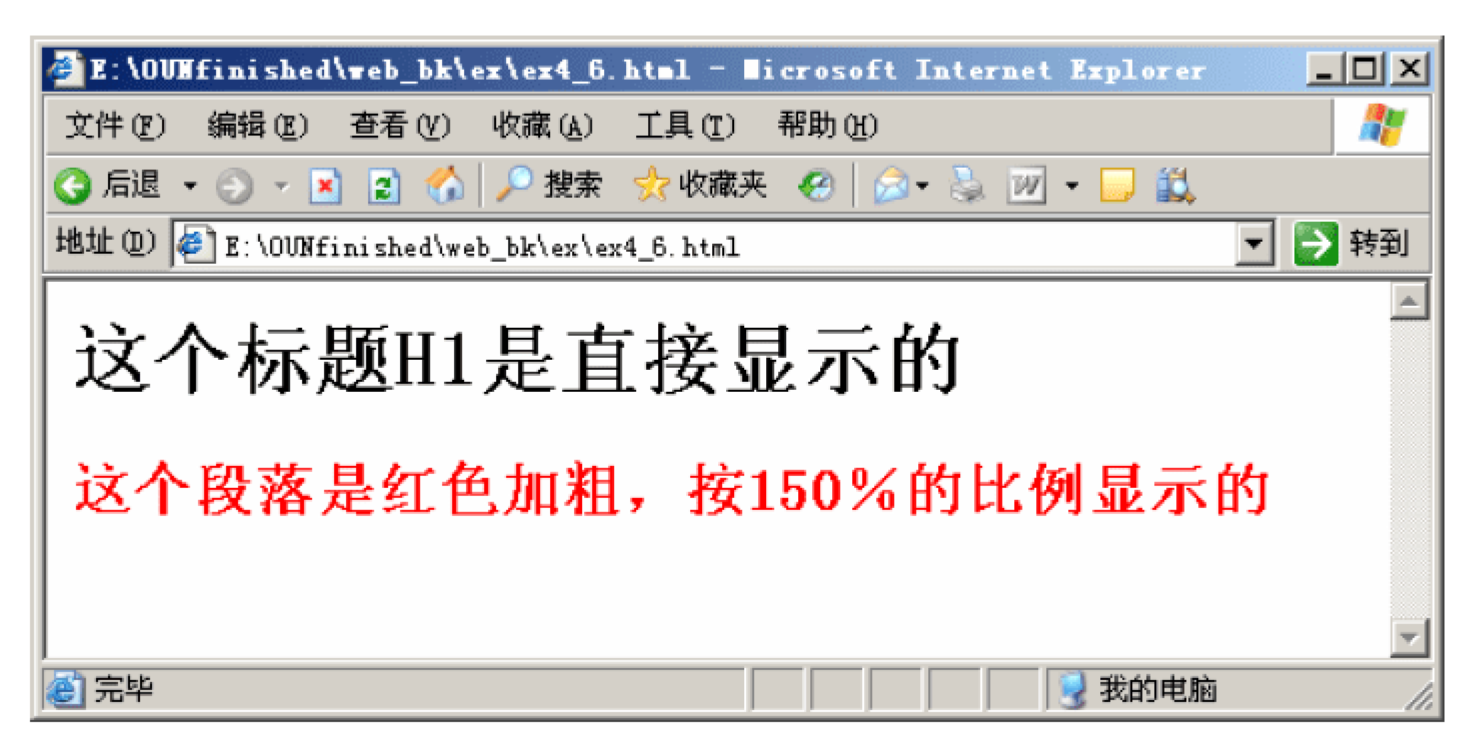


图 5-6　使用限制匹配的 ID 选择符

注意：

由于 ID 选择符或类选择符有一定的局限性，它们能单独定义某个元素的样式或者不限定元素而成为通用样式，因此一般会在一些特殊的情况下使用，或根据需要选择使用。

5.3.5　包含选择符

包含选择符是一种单独针对某种元素的包含关系而定义的样式表。假设元素 1 里包含元素 2，针对这种结构定义的包含选择符对单独的元素 1 或元素 2 无影响，例如：

```
table a
{
```

```
    font-size: 32px
}
```

【实例 5-7】包含选择符的使用

程序代码如 ex5_7.html 所示。

ex5_7.html

```
<HTML>
  <HEAD>
    <STYLE>
      table a
      {
        font-size: 32px
      }
    </STYLE>
  </HEAD>
  <BODY>
    <TABLE border>
      <CAPTION>张三销售业绩</CAPTION>
      <TR><TH>编号</TH><TH>姓名</TH><TH>外销</TD><TH>内销</TH><TH>总数</TH>
      <TR><TD>0001</TD><TD><a href="http://www.sohu.com">张　三
</a></TD><TD>45</TD><TD>86</TD><TD>131</TD>
    </TABLE>
    <br>
    <a href="http://www.sohu.com">表格外的链接</a>
  </BODY>
</HTML>
```

在本实例中，表格内的链接样式被改变了，其文字大小为 32 像素，而表格外链接的文字仍为默认大小，该实例运行后的浏览器显示如图 5-7 所示。



图 5-7　使用包含选择符

5.3.6　样式表的层叠性

层叠性指的是继承性，样式表的继承规则是外部的元素样式继承给这个元素所包含的其他元素。实际上，所有嵌套在元素中的元素都会继承外层元素已指定的属性值，有时会把很多层次所嵌套的样式叠加在一起，除非另外设置。例如：在 DIV 标记中嵌套了 P 标记，如下所示：

```
DIV { color: red; font-size:9pt}
…
<div>
   <p>
      这个段落的文字为红色 9 号字
   </P>
</div>
```

此处，P 元素里的内容会继承 DIV 中所定义的样式。有时内部选择符不继承周围选择符的值，但理论上这些都是特殊的。例如，上边界属性值是不会继承的，直觉上，一个段落不会具有同文档 BODY 一样的上边界值。

另外，当样式表在继承的过程中遇到冲突时，总是以最近定义的样式为准。

【实例 5-8】样式表的继承

程序代码如 ex5_8.html 所示。

ex5_8.html

```
<HTML>
   <HEAD>
      <STYLE>
         DIV{ color: red; font-size:9pt}
         P{color: blue}
      </STYLE>
   </HEAD>
   <BODY>
      <div>
         <p>
            这个段落的文字为蓝色 9 号字
         </p>
      </div>
   </BODY>
</HTML>
```

在这个例子中，可以看到段落里的文字大小为 9 号字，这是由 DIV 继承而得到的；而 color 属性则依照<P>标签中最后定义的样式。该实例运行后的浏览器显示如图 5-8 所示。

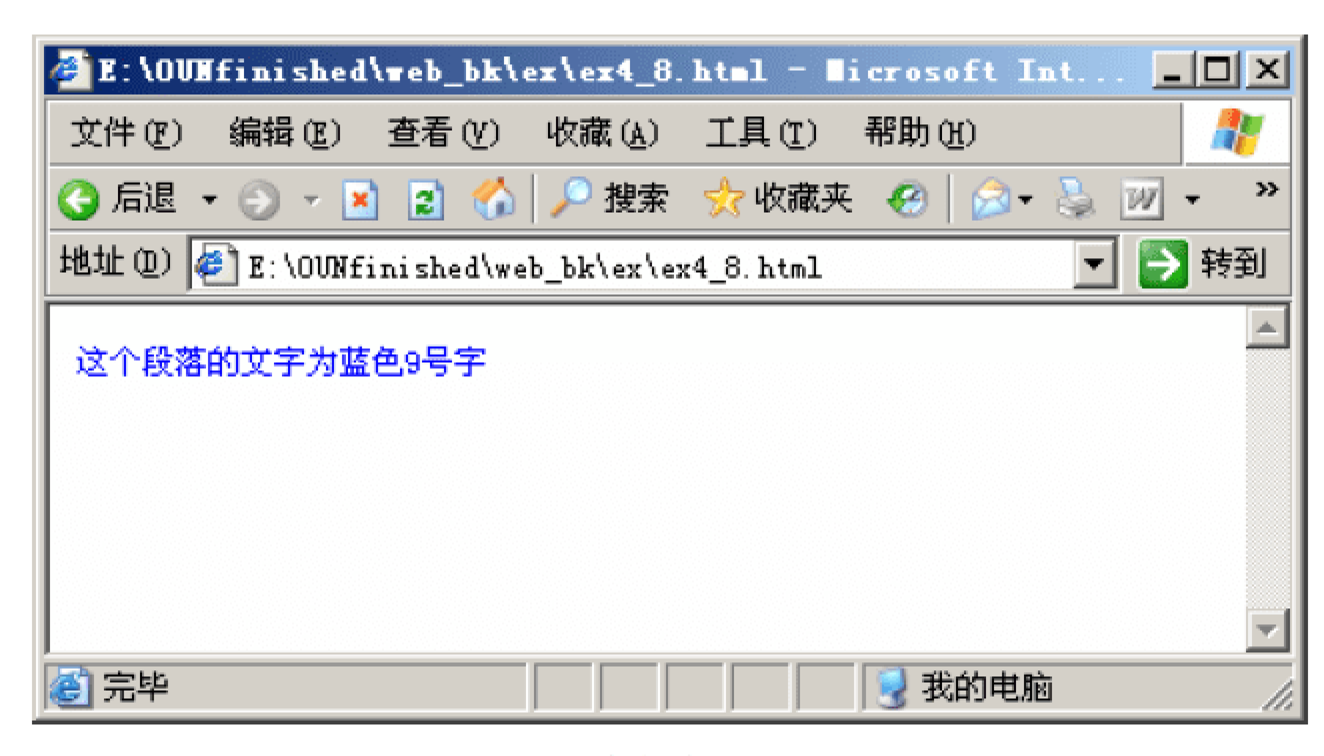


图 5-8　样式表的继承

不同的选择符定义相同的元素时，要考虑到不同的选择符之间的优先级。ID 选择符、类选择符和 HTML 标记选择符，因为 ID 选择符是最后施加到这个元素上的，所以优先级

最高，其次是类选择符。如果想改变这三者之间默认的关系，可以用!important 来提升某个样式表的优先权，例如：

```
p { color: #FF0000!important }
```

【实例 5-9】选择符的优先级

程序代码如 ex5_9.html 所示。

ex5_9.html

```
<HTML>
  <HEAD>
    <STYLE>
      p{color:red!important}
      .blue{color:blue}
      #id1{color:green}
    </STYLE>
  </HEAD>
  <BODY>
    <div>
      <p id="id1" class="blue">
        这个段落的文字为红色字
      </p>
    </div>
  </BODY>
</HTML>
```

在这个例子中，对页面中同一个段落使用了三种样式，浏览器最后会按照由!important 标明的 HTML 标签选择符来设定，因此显示为红色文字。如果去掉!important，则依据优先权最高的 ID 选择符而显示为绿色文字，该实例运行后的浏览器显示如图 5-9 所示，请读者修改后再进行一次测试，以便比较效果。

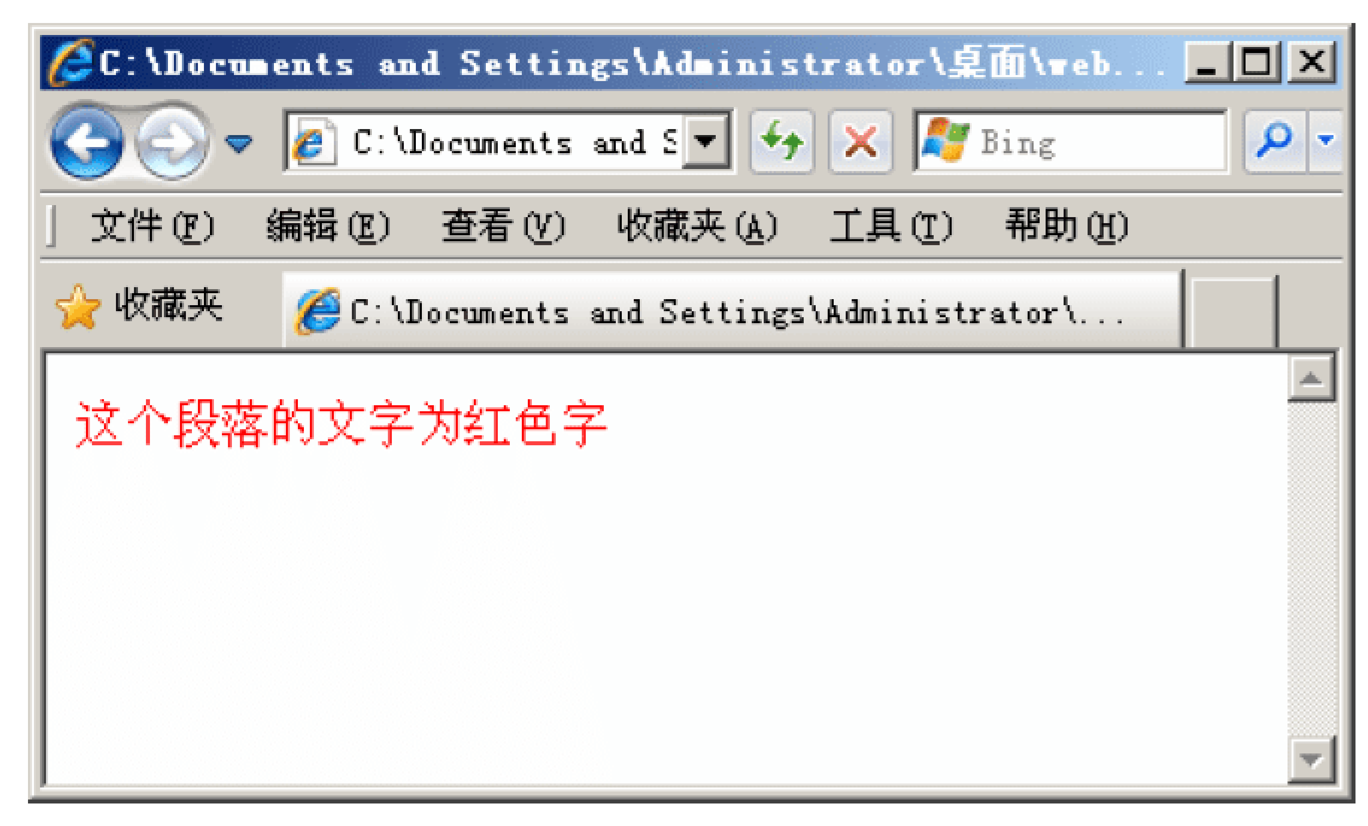


图 5-9　选择符的优先级

5.3.7　伪类

除了类型选择符、ID 选择符和类选择符之外，CSS 也允许使用伪类和伪元素选择符。

伪类选择符是一组基于预定义性质的选择符，HTML 元素可以使用这些预定义性质。实际上这些性质与 class 属性的功能是相同的，因此在 CSS 术语中，它们被称作伪类。对

应这些伪类的标签，其中不存在真正的 class 属性；相反，它们代表应用到这些元素的某些方面，或者是相对于该元素的浏览器用户界面的状态。

伪类可以看作一种特殊的类选择符，一种支持 CSS 的浏览器能自动识别的特殊选择符。其最大的用处在于可以对链接在不同状态下定义不同的样式效果。伪类的语法在原有的语法里加上一个伪类(pseudo-class)说明，定义方法如下：

```
selector:pseudo-class {property: value}
```

由上面代码可得伪类的格式为选择符:伪类 {属性:值}。伪类和类不同，它是在 CSS 中已经定义好的，表 5-1 列出了这些伪类。它们不能像类选择符一样随意用别的名字，根据上面的语法可以解释为对像(选择符)在某个特殊状态下(伪类)的样式。类选择符及其他选择符也同样可以和伪类混用，其定义方法如下：

```
selector.class:pseudo-class {property: value}
```

类选择符及其他选择符合伪类混用格式为选择符.类:伪类 {属性: 值}。

表 5-1　CSS 中的伪类

伪　类　名	说　　明
:active	被激活的元素(在鼠标单击与释放之间发生的事件)时的样式
:first-child	元素的第一个子对象的样式
:first	设置页面容器第一页使用的样式
:focus	设置对象在成为输入焦点(该对象的 onfocus 事件发生，如接收输入的表单)时的样式
:hover	设置对象在其鼠标悬停时的样式
:lang()	设置对象使用特殊语言的内容的样式
:left	设置页面容器位于装订线左边的所有页面使用的样式
:link	设置 a 元素在未被访问前的样式
:right	设置页面容器位于装订线右边的所有页面使用的样式
:visited	设置 a 对象在其链接地址已被访问过时的样式

以下对几种常用的伪类进行说明和介绍。

1. 锚的伪类

锚的伪类是最常用的是 4 种 a(锚)元素的伪类，它表示动态链接在 4 种不同的状态：link、visited、active、hover(未访问的链接、已访问的链接、激活链接和鼠标停留在链接上)。可以把它们分别定义为不同的效果。

```
a:link {color: #FF0000; text-decoration: none} /* 未访问的链接 */
a:visited {color: #00FF00; text-decoration: none} /* 已访问的链接 */
a:hover {color: #FF00FF; text-decoration: underline} /* 鼠标停留在链接上 */
a:active {color: #0000FF; text-decoration: underline} /* 激活链接 */
```

【实例 5-10】锚的伪类

程序代码如 ex5_10.html 所示。

ex5_10.html

```
<HTML>
  <HEAD>
    <STYLE>
      a:link {color: #FF0000; text-decoration: none} /*  未访问的链接  */
      a:visited {color: #00FF00; text-decoration: none} /*  已访问的链接  */
      a:hover {color: #FF00FF; text-decoration: underline} /*  鼠标停留在链接上  */
      a:active {color: #0000FF; text-decoration: underline} /*  激活链接  */
    </STYLE>
  </HEAD>
  <BODY>
    <a href="http://www.sohu.com">去搜狐首页</a>
  </BODY>
</HTML>
```

在本实例中，这个链接未访问时的颜色是红色并无下划线，访问后是绿色并无下划线，激活链接时为蓝色并有下划线，鼠标停留在链接上时为紫色并有下划线。请读者自己在浏览器中运行这个实例来观察实际的效果。

技巧：

有时这个链接访问前鼠标指向链接时有效果，而链接访问后鼠标再次指向链接时却没有效果了。这可能是因为定义时将 a:hover 放在了 a:visited 的前面，这样的话由于后面的优先级高，当访问链接后就忽略了 a:hover 的效果。所以根据叠层顺序，在定义这些链接样式时，需要按照 a:link、a:visited、a:hover、a:active 的顺序书写。

2. 伪类和类选择符

将伪类和类组合起来用，就可以在同一个页面中做几组不同的链接效果了。

【实例 5-11】伪类和类选择符

程序代码如 ex5_11.html 所示。

ex5_11.html

```
<HTML>
  <HEAD>
    <STYLE>
      a.red:link {color: #FF0000}
      a.red:visited {color: #0000FF}
      a.blue:link {color: #00FF00}
      a.blue:visited {color: #FF00FF}
    </STYLE>
  </HEAD>
  <BODY>
```

```
        <a class="red" href="...">这是采用 red 类的一组链接</a>
        <br>
        <a class="blue" href="...">这是采用 blue 类的一组链接</a>
    </BODY>
</HTML>
```

该例利用伪类和类选择符定义了两组链接，其中一组为红色，访问后为蓝色；另一组为绿色，访问后变为紫色。请读者自行运行此实例以观察实际的效果。

5.3.8　伪对象

样式表中定义了 4 种伪元素，即虚拟元素，它们是根据内容创建的，是与基本元素相关的，这些元素如表 5-2 所示。

表 5-2　CSS 中的伪元素

伪元素名	说　明
:after	用来和 content 属性一起使用，设置在对象后(依据对象树的逻辑结构)发生的内容
:before	用来和 content 属性一起使用，设置在对象前(依据对象树的逻辑结构)发生的内容
:first-letter	此伪元素仅作用于块对象。内联要素要使用该属性，必须先设定对象的 height 或 width 属性，或者设定 position 属性为 absolute，或者设定 display 属性为 block。设置对象内的第一行样式
:first-line	此伪元素仅作用于块对象。内联要素要使用该属性，必须先设定对象的 height 或 width 属性，或者设定 position 属性为 absolute，或者设定 display 属性为 block。如果未强制指定对象的 width 属性，首行的内容长度可能不是固定的

伪元素:before 和:after 用于插入已产生的内容；:first-letter 和:first-line 可以对元素的首字或首行设定不同的样式，请读者参看下面的实例。

【实例 5-12】首字和首行的伪类

程序代码如 ex5_12.html 所示。

ex5_12.html

```
<HTML>
    <HEAD>
        <STYLE>
            p:first-letter {font-size: 300%}
            div:first-line {color: red;font-size:200%}
        </STYLE>
    </HEAD>
    <BODY>
        <p>
          这是一个段落，这个段落的首字被放大了。
        </p>
        <div>
              这是段落的第一行<br>
```

```
            这是段落的第二行<br>
            这是段落的第三行<br>
        </div>
    </BODY>
</HTML>
```

在这个例子中，对段落标签<P>定义了文本首字尺寸为默认大小的 3 倍，对<DIV>标签定义了第一行为红色且字体为默认大小的 2 倍，而第二、第三行为默认颜色，运行后的浏览器显示如图 5-10 所示。需要注意的是，首字和首行的伪类需要 IE5.5 以上的版本支持。

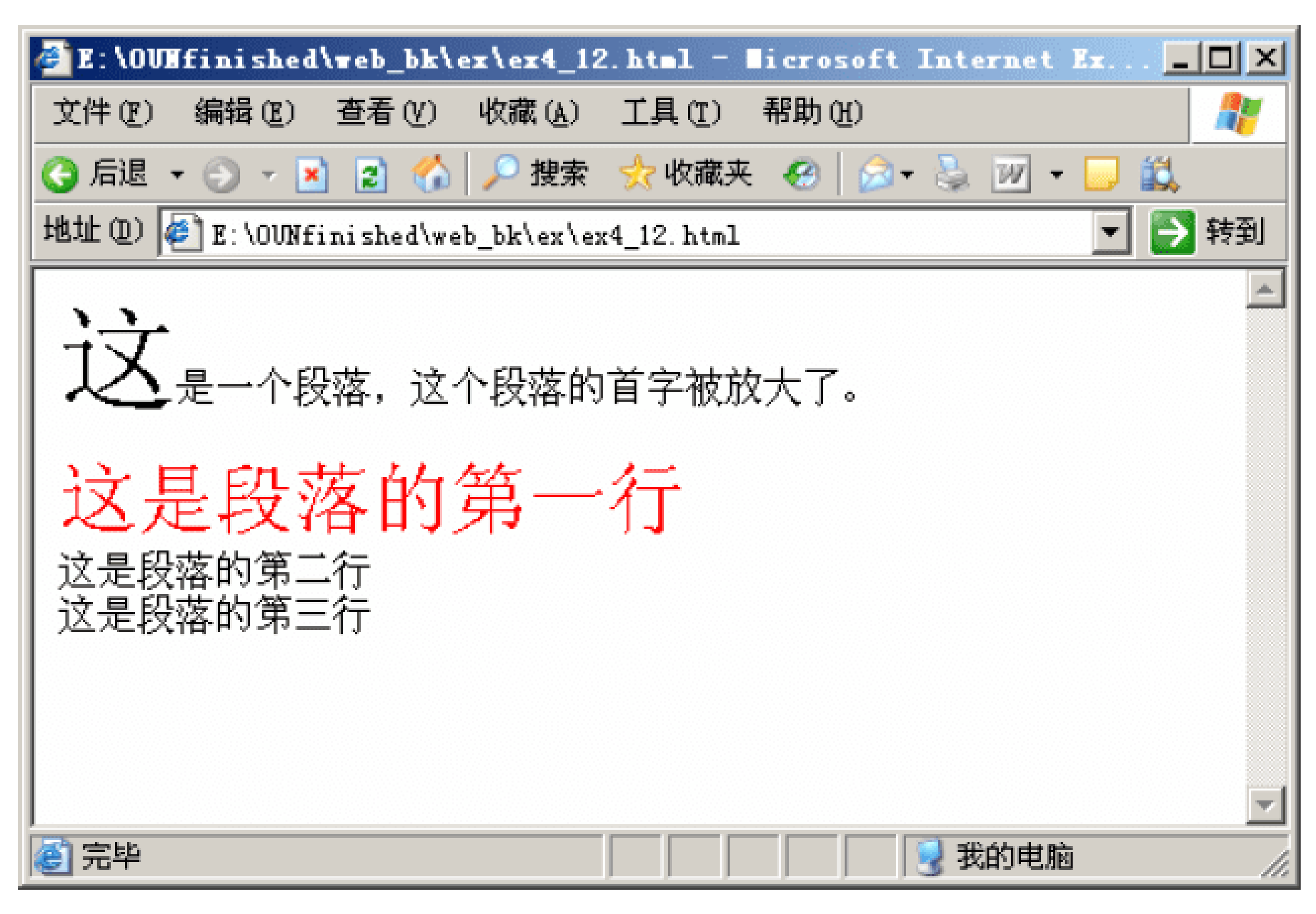


图 5-10　首字和首行的伪类

5.3.9　注释

可以在 CSS 中插入注释来说明代码的含义，注释有利于自己或别人今后在编辑和更改代码时理解代码的含义；在浏览器中，对于注释部分的内容是被忽略的因而不会产生实际效果。和 HTML 的注释方式不同，CSS 注释以“/*”开头，以“*/”结尾，表示如下：

```
/* 定义段落样式表 */
p
{
   text-align: center; /* 文本居中排列 */
   color: black; /* 文字为黑色 */
   font-family: arial /* 字体为 arial */
}
```

5.3.10　对 DIV+CSS 方案的思考

对于网页布局方案各人有不同的选择和偏好，比如有人喜欢采用表格(TABLE)来完成整个网页的布局设置，而现在一种流行的方式是采用 DIV 标签与 CSS 组合。采取这种技术方案，相对而言具有以下几个方面的优点和缺点。

1. 优点

(1) 有利于搜索引擎爬虫程序

一般而言，相同网页页面 HTML 文件 TABLE 布局字节大于 DIV+CSS 布局的字节，因此这种方案可以提高搜索引擎爬虫爬行和下载页面的效率。

(2) 重构或改版时，修改较为方便

一般 DIV+CSS 页面都是 HTML 和 CSS 文件分开的，也就是一个网页的内容与表现形式是分离的，一般修改 CSS 文件中的 CSS 样式属性就可以达到修改整个网站的样式的效果，如背景颜色、字体颜色、网站宽度等。在这方面，TABLE 方案如果没有采用分离 CSS 文件的策略则往往显得没这么方便。

(3) DIV+CSS 页面增加网页打开速度

技术方案的特性决定了其性能，因为 DIV+CSS 通常是 DIV 的 HTML 和 CSS 文件分开的，而浏览器打开该网页的时候是同时下载 HTML 和 CSS 的，所以相应提高了网页打开的速度。TABLE 的特性是浏览器打开的时候必须是浏览器下载以< TABLE >开始，并以</ TABLE >结束的所有内容之后才显示该块的内容，而 DIV 的 HTML 是边加载边将内容呈现到浏览器上，因此 DIV+CSS 方案的网站能大大增强用户体验。大家都知道网页多等 1 秒钟都会降低用户等待时间。解析谷歌将网页加载速度快慢作为影响排名重要因素.

采用 DIV+CSS 解决方案可能存在的缺陷：

(1) 开发技术要求较高

对开发者的技术水平提出了较高要求，在兼容各浏览器及版本浏览器方面也更加困难。

(2) 开发时间长

由于技术复杂，因此 DIV+CSS 布局相对 TABLE 布局所需要的开发和制作时间会更长一些。

(3) 开发成本相对 TABLE 方案更高

因为技术性及时间方面的特性，最终决定了 DIV+CSS 方案比 TABLE 方案的总体成本要略高。

5.4　CSS 的滤镜及其应用

相对于纯文本，视觉和听觉的感受有时更容易带来震撼的效果，因此利用多媒体手段来表达可以丰富网页的展示形式。CSS 除了能对 HTML 的显示样式进行定义和管理以外，也提供了多媒体处理方面的滤镜功能。CSS 的滤镜能利用客户端的计算资源对图片等资源生成类似于 Photoshop 特效滤镜的处理效果。虽然 CSS 所带的滤镜种类比 Photoshop 要少很多，但对于网页应用而言，已经能满足大多数应用的需求了。

CSS 滤镜属性的标识符是 filter，其定义方式为：

```
filter：filtername(parameters)
```

filter 是滤镜属性选择符，只要利用滤镜操作，就必须先定义 filter；filtername 是滤镜属性名，可以使用包括 alpha、blur、chroma 等在内的多种属性，属性的后面还可以增加一些说明参数。

注意：

由于 Internet Explorer 4.0 版本才开始提供 CSS 的滤镜功能，因此，该版本以前的浏览器是无法看到本节所介绍的效果的；且由于 4.0 版本只支持少数几个滤镜，因此建议使用更高的版本，另外对于不同浏览器会对滤镜作出不同的定义，如果需要支持主流的多种浏览器，就需要额外的处理，可参看下文的叙述。

CSS 滤镜按其功能可分为：界面滤镜(Procedural Surfaces)、静态滤镜(Static filters)和转换滤镜(Transitions)三种，以下分别进行介绍。

5.4.1　界面滤镜

界面滤镜是一个显示在对象内容和对象背景之间彩色的层，它可以动态定义每个像素点的颜色和 Alpha 值。主要包括 AlphaImageLoader 和 Gradient 两种。

1. 滤镜 AlphaImageLoader

使用该滤镜在对象容器边界内，可以在对象的背景和内容之间显示一张图片，并提供对此图片的剪切和改变尺寸的操作，其使用方法如下：

```
filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(src='image.PNG', sizingMethod ='scale');
```

其中属性“sizingMethod”可以定义为 image、scale 或 crop，三种定义的含义通过下面的例子可以说明。

【实例 5-13】滤镜 AlphaImageLoader 的用法

程序代码如 ex5_13.html 所示。

ex5_13.html

```
<HTML>
  <HEAD>
    <TITLE>AlphaImageLoader 演示</TITLE>
<style>
  #png_image{
  background: none;
  position: absolute;
  left: 0;
  top: 0;
  width: 200px;
  height: 200px;
  filter: progid:DXImageTransform.Microsoft.AlphaImageLoader(src='image.PNG',sizingMethod='scale');
}
</style>
  </HEAD>
  <BODY>
    <div id="png_image">如果载入的是 PNG(Portable Network Graphics)格式的图片，则可以提供
0%~100%的透明度。PNG(Portable Network Graphics)格式的图片的透明度不妨碍你选择文本。也就是说，
你可以选择显示在 PNG(Portable Network Graphics)格式的图片完全透明区域后面的内容
    </div>
```

```
  </BODY>
</HTML>
```

该实例运行后的浏览器显示如图 5-11 所示，在这个例子中，使用了滤镜 AlphaImageLoader。其中最左边的图为上面的代码所生成的效果，由于将属性 sizingMethod 设置为“scale”，因此图片被放大了；中间的图将属性 sizingMethod 设置为“crop”，因此图片按照其自身大小来显示；右边的图是将属性 sizingMethod 设置为“image”，因此图片大小以外的文字没有显示。此外，这个例子在 IE 中显示时，用鼠标选择时，只能选择到文字，图片在这里是无法选择到的。

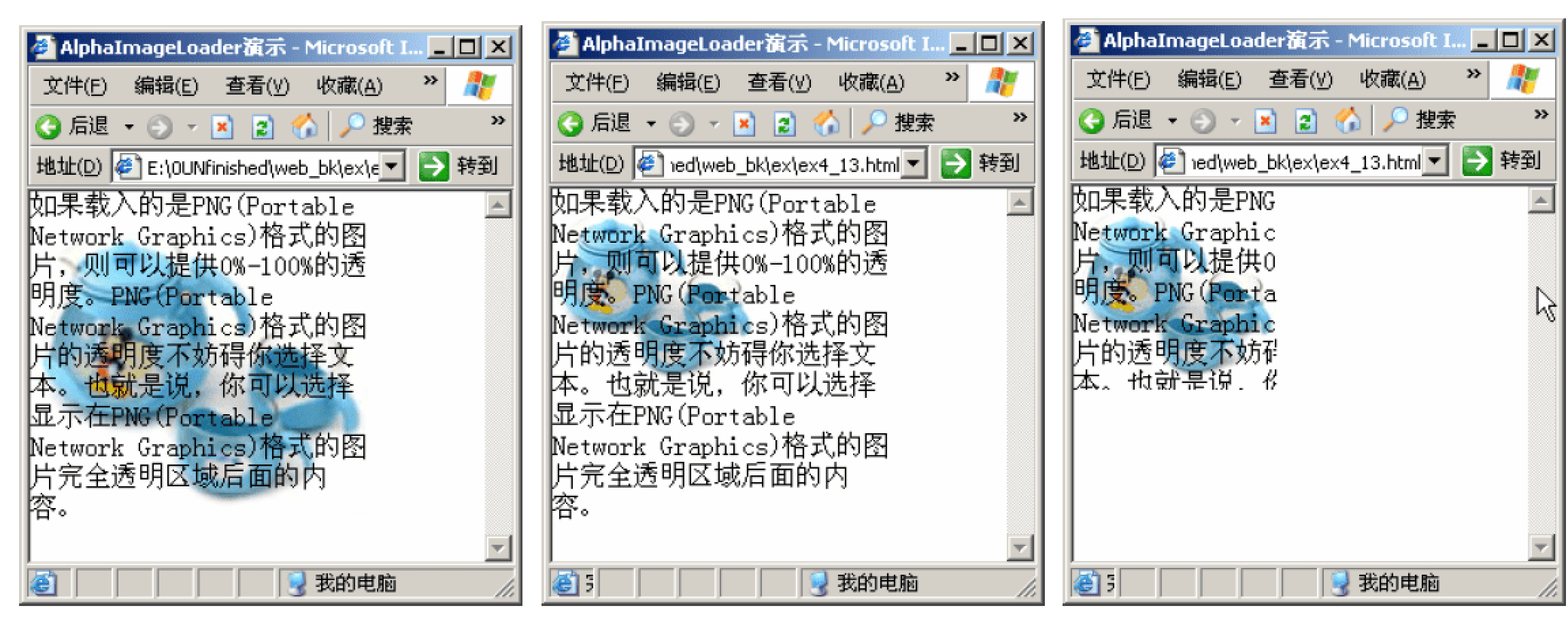


图 5-11　滤镜 AlphaImageLoader 的用法

如果载入的是 PNG(Portable Network Graphics)格式的图片，则可以提供 0%~100%的透明度选项。PNG 格式的图片的透明度不妨碍选择文本。也就是说，用户可以选择显示在 PNG 格式的图片完全透明区域后面的内容。

2. 滤镜 Gradient

使用 Gradient 滤镜，可以在对象的背景和内容之间显示定制的色彩层。当此效果通过转变显示时，在渐变色彩层之上的文本程序性的初始化为透明的，当色彩渐变实现后，文本颜色会以其定义的值更新，其使用方法如下：

```
filter:progid:DXImageTransform.Microsoft.Gradient(GradientType=1,StartColorStr='#00FF00',
EndColorStr='#FFFFFF');
```

其中“GradientType”属性控制了色彩渐变的方向。当它定义为 0 时，色彩将从上向下渐变；定义为 1 时，从左向右。

【实例 5-14】滤镜 Gradient 的用法

程序代码如 ex5_14.html 所示。

ex5_14.html

```
<HTML>
  <HEAD>
    <TITLE>滤镜 Gradient 的用法</TITLE>
    <STYLE>
```

```
            #gradient1
            {
              font:18   pt  宋体;
              color:white;
              filter:progid:DXImageTransform.Microsoft.Gradient(GradientType=1, StartColorStr='#000000',
EndColorStr='#FFFFFF');
            }
          </STYLE>
        </HEAD>
        <BODY id="gradient1">
          使用方法：filter:progid:DXImageTransform.Microsoft.Gradient(GradientType=1,
StartColorStr='#000000', EndColorStr='#FFFFFF');
        </BODY>
      </HTML>
```

该实例运行后的浏览器显示如图 5-12 所示，本例中使用滤镜 Gradient 生成了渐变的背景色。

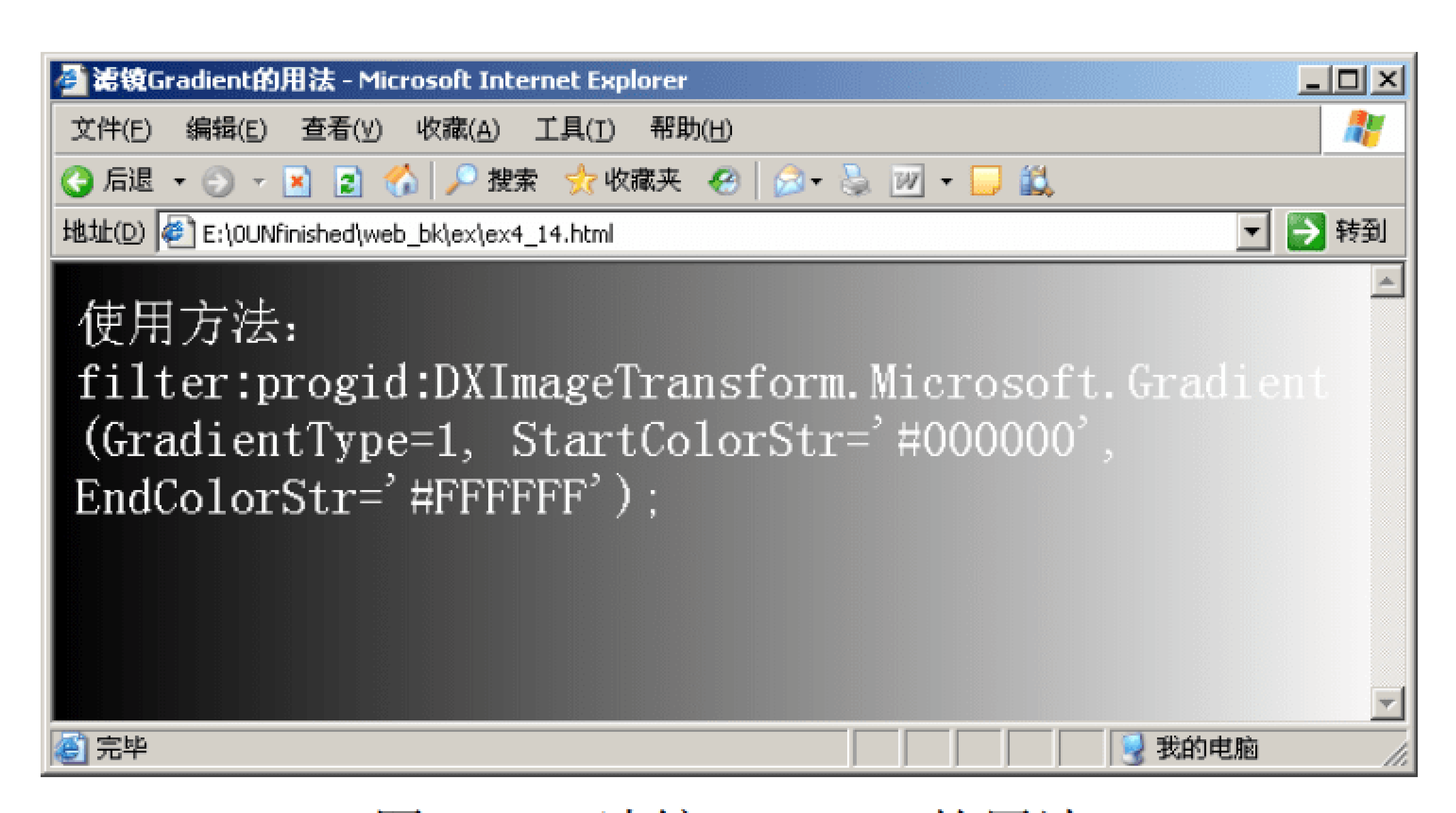


图 5-12　滤镜 Gradient 的用法

5.4.2　静态滤镜

静态滤镜是通过静态的方式改变某个对象内容的显示效果。关于常用静态滤镜的属性及其说明如表 5-3 所示。

表 5-3　静态滤镜属性及其说明

滤镜属性	说　明
Alpha	调整对象内容的透明度，可以设置整体透明度，或线性渐变和放射渐变的透明度
BasicImage	用于色彩处理、图像旋转，或对象内容的透明度；当此效果通过转变显示时，在渐变色彩层之上的文本程序性地初始化为透明的，当色彩渐变实现后，文本颜色会以其定义的值更新
Blur	制作对象内容的模糊效果
Chroma	将对象中指定的颜色显示为透明；此效果在羽化(柔化色彩以同周围相邻的颜色平和的过渡)的线条等处也不会很好地作用；确定的 color 参数值可能会导致图片自身的透明颜色变为不透明
Compositor	依据初始对象和新对象色彩的特定混合方式显示新的对象内容；这个滤镜提供了丰富的将输入对象的色彩和透明度相互作用的图像合成的功能设置

(续表)

滤镜属性	说　明
DropShadow	制作对象的阴影效果
Emboss	用灰度值为对象内容制作浮雕纹理效果(凸出)
Engrave	用灰度值为对象内容制作浮雕纹理效果(凹下)
Glow	环绕对象内容边缘添加辉光制作发热效果。辉光将出现在对象边界内的内容的最外轮廓之外
Light	为对象的内容建立光照效果
MaskFilter	将对象内容的透明像素用 color 参数指定的颜色显示作为一个遮罩，而非透明像素则转为透明
Matrix	使用矩阵变形实现对象内容的改变尺寸、旋转、上下或左右反转
MotionBlur	为对象内容制作运动模糊效果
Shadow	为对象内容建立阴影效果
Wave	为对象内容建立波纹扭曲效果
ICMFilter	根据颜色配置文件(.icm)转换对象的色彩内容。这样能激活对某些细节内容的显示改良，或对硬件设备输出的模拟显示，如打印机或显示器
Gray	灰度显示对象内容
Invert	反相显示对象内容
Xray	以 X 光效果显示对象内容
FlipH	水平翻转对象内容
FlipV	垂直翻转对象内容

以下对常用的滤镜进行简要的介绍，读者可以参考这些基本的方法，自行选择所需要的特效。

1. Alpha 滤镜

这个滤镜的功能为：使对象产生透明度，其基本语法为：

```
filter: progid:DXImageTransform.Microsoft.Alpha (enabled=bEnabled, style=iStyle, opacity=iOpacity, finishOpacity=iFinishOpacity, startX=iPercent, startY=iPercent, finishX=iPercent,    finishY=iPercent )
```

其中的参数说明如表 5-4 所示。

表 5-4　滤镜 Alpha 的参数及其说明

参 数 名 称	功 能 描 述	参数取值说明
Opacity	图片的不透明度	值为 0～100，0 为完全透明，100 为完全不透明
FinishOpacity	设置渐变的透明效果时，用来指定结束时的透明度	值为 0～100，0 为完全透明，100 为完全不透明
Style	指定渐变的显示形状	0：没有渐进；1：直线渐进；2：圆形渐进；3：矩形渐进

(续表)

参 数 名 称	功 能 描 述	参数取值说明
StartX	渐变透明效果开始的 X 坐标值	
StartY	渐变透明效果开始的 Y 坐标值	
FinishX	渐变透明效果结束的 X 坐标值	
FinishY	渐变透明效果结束的 Y 坐标值	

下面的例子说明了 Alpha 滤镜的用法。

【实例 5-15】Alpha 滤镜的使用

程序代码如 ex5_15.html 所示。

ex5_15.html

```
<HTML>
  <HEAD><TITLE>Alpha 演示</TITLE>
  </HEAD>
  <BODY>
    <IMG SRC="image.jpg" WIDTH="150">
    <IMG SRC="image.jpg" WIDTH="150"
      STYLE="filter:alpha(opacity:60)">
    <IMG SRC="image.jpg" WIDTH="150"
      STYLE="filter:alpha(opacity:60, style:1)">
  </BODY>
</HTML>
```

在这个例子中，对同一张图片设置了不同的 Alpha 滤镜。左边为原始图片；中间的设置了 opacity:60，即不透明度为 60%；而右边的设置为 opacity:60, style:1，即在中间图片的基础上增加了直线渐进的设置。图 5-13 为在浏览器中的显示效果。

注意：

虽然在浏览器中可以看到经滤镜处理后的效果，但如果直接使用鼠标右键单击图片选择“图片另存为(S)”后，所得到的图片仍然是原始的图片。

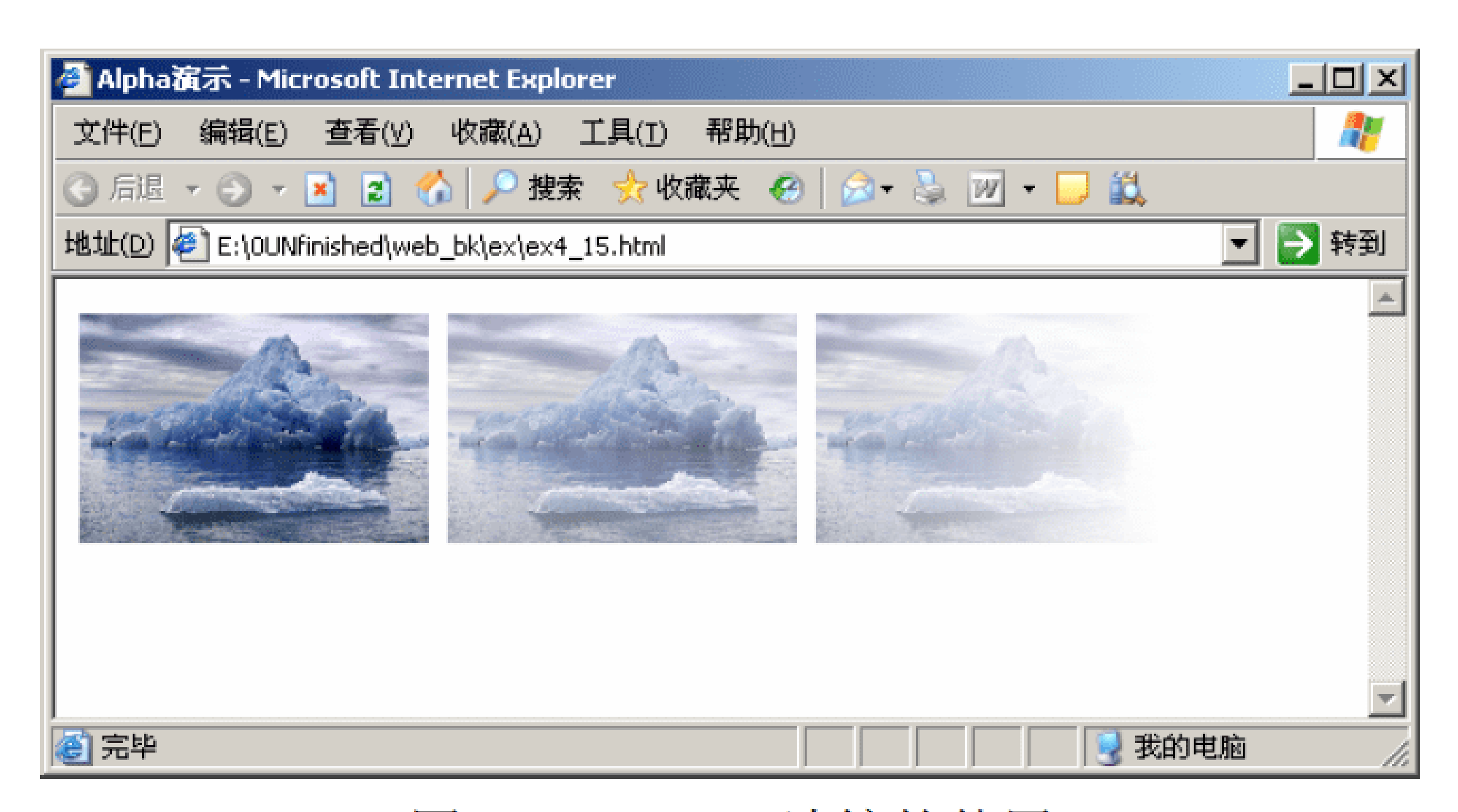


图 5-13　Alpha 滤镜的使用

不同浏览器对于滤镜的描述是存在差异的，例如：

```
IE：filter:alpha(opacity=60)
FireFox：-moz-opacity: 0.6
Chrome：opacity: 0.6
```

所以对于一个网页需要同时兼容三种浏览器，则需要将这个 CSS 样式的代码写为：

```
.test{
filter:alpha(opacity=60);
-moz-opacity: 0.6;
opacity: 0.6;
}
```

这样做既能在 IE 中看到，也能在 Chrome 和 Firefox 中看到，至于其他浏览器目前通常就可以忽略，当然如果觉得必要也可以加上相应的代码。此外一些基于 IE 内核的浏览器，则只要 IE 里正常则不会出现问题。

2. Blur 滤镜

这个滤镜可以制作模糊的特效，其基本用法如下：

```
filter:progid:DXImageTransform.Microsoft.Blur(enabled=bEnabled,makeShadow= bShadow ,
pixelRadius=flRadius, shadowOpacity=fOpacity)
```

常用 Blur 滤镜的参数名称、功能描述及其参数取值说明如表 5-5 所示。

表 5-5　滤镜 Blur 的参数及其说明

参 数 名 称	功 能 描 述	参数取值说明
enabled	设置或检索滤镜是否激活	ture：激活；false：禁止
makeShadow	设置或检索对象的内容是否被处理为阴影显示	ture：是；false：否，选择为是，模糊效果不再出现
pixelRadius	设置或检索模糊效果的作用深度	数值，越大则越模糊
shadowOpacity	设置或检索使用 makeShadow 制作成的阴影的透明度(暗度)	0～1 之间；1 为完全不透明

【实例 5-16】Blur 滤镜的使用

程序代码如 ex5_16.html 所示。

ex5_16.html

```
<HTML>
  <HEAD>
    <TITLE>滤镜 Blur 的用法</TITLE>
  </HEAD>
  <BODY>
    <IMG SRC="image.jpg" WIDTH="150">
    <IMG SRC="image.jpg" WIDTH="150" STYLE="filter:progid:
        DXImageTransform.Microsoft.Blur(makeShadow=false,pixelRadius=3,shadowOpacity=0);">
  </BODY>
</HTML>
```



图 5-14　Blur 滤镜的使用

在这个例子中，浏览器中的显示效果如图 5-14 所示，其中，左边的图为原图，右边的图片为设置了 Blur 滤镜后的效果。

3. MotionBlur 滤镜

这个滤镜与上面提到的 blur 不同，它使得对象内容产生运动模糊效果。定义方式如下：

```
filter:progid:DXImageTransform.Microsoft.MotionBlur(enabled=bEnabled, add=bAddImage, direction=iOffset, strength=iDistance)
```

其主要参数说明如表 5-6 所示。

表 5-6　滤镜 Motion Blur 的参数及其说明

参 数 名 称	功 能 描 述	参数取值说明
Add	指定图片是否显示原来的模糊方向	0：不显示原对象；1：显示原对象
Direction	设置模糊的方向	0(上)，45(右上)，90(右)，135(右下)，180(下)，225(左下)，270(左)，315(左上)
Strength	指定模糊图像模糊的半径大小	以 pixels 为单位，默认为 5

【实例 5-17】MotionBlur 滤镜的使用

程序代码如 ex5_17.html 所示。

ex5_17.html

```
<HTML>
  <HEAD>
    <TITLE>滤镜 MotionBlur 的用法</TITLE>
  </HEAD>
  <BODY>
  <IMG SRC="image.jpg" WIDTH="150">
  <IMG SRC="image.jpg" WIDTH="150" STYLE="filter:progid:
        DXImageTransform.Microsoft.MotionBlur(add=false,direction=135,strength=8);">
  </BODY>
</HTML>
```



图 5-15　MotionBlur 滤镜的使用

在这个例子中，浏览器中的显示效果如图 5-15 所示，其中，左边的图为原图，右边的图片为设置了 MotionBlur 滤镜后的效果。

4. DropShadow 滤镜

该滤镜用于生成对象的阴影效果，此滤镜可以应用于图片，但一般用于制作文字阴影，其定义方式如下：

```
filter:progid:DXImageTransform.Microsoft.DropShadow(enabled=bEnabled, color=sColor, offX=iOffsetX,
offY=iOffsetY, positive=bPositive)
```

其主要参数说明如表 5-7 所示。

表 5-7　滤镜 DropShadow 的参数及其说明

参 数 名 称	功 能 描 述	参数取值说明
Color	指定阴影的颜色	RGB 格式的颜色值
OffX	指定阴影相对于对象在水平方向的偏移	整数。正数表示阴影在对象右方，负数表示在左方
OffY	指定阴影相对于对象在水平垂直方向的偏移	整数。正数表示阴影在对象上方，负数表示在下方

【实例 5-18】DropShadow 滤镜的使用

程序代码如 ex5_18.html 所示。

ex5_18.html

```
<HTML>
  <HEAD>
    <TITLE>滤镜 DropShadow 的用法</TITLE>
  </HEAD>
  <BODY>
    <IMG SRC="ray.gif" WIDTH="92">
    <IMG SRC="ray.gif" WIDTH="92" STYLE="filter:progid:
      DXImageTransform.Microsoft.DropShadow(color=#FF404040,offX=3,offY=3,positives=false);">
  </BODY>
</HTML>
```

在这个例子中，浏览器中的显示效果如图 5-16 所示。其中左边的图为原图，右边的图片为设置了 DropShadow 滤镜之后的效果。



图 5-16　DropShadow 滤镜的使用

注意：

上面重点介绍了几种滤镜的使用方法，读者可以找到一份关于 CSS 滤镜的说明文档，其中采用类似于字典的方式说明了所有滤镜的用法和每个参数的含义。此处通过这几个典型滤镜用法的说明，是希望能引导读者了解说明文档的含义和基本用法。因此，对于其他更多滤镜的使用方法，请读者根据上述的方法自行验证和测试。

5.4.3　转换滤镜

转换滤镜主要是用来处理网页或是 HTML 元素对象显示效果的，它可以在新旧内容显示交替转换时产生特定的视觉效果，常用的转换滤镜的属性及其说明如表 5-8 所示。

表 5-8　转换滤镜属性及其说明

滤镜属性	说　明
Barn	用模拟开关门效果转换对象内容
BlendTrans	用渐隐效果转换对象内容
Blinds	用百叶窗开关效果转换对象内容
Checker Board	用类似国际象棋棋盘的网格推拉效果转换对象内容
Fade	用渐隐效果转换对象内容
GradientWipe	用滚动渐隐效果转换对象内容
Inset	用对角扩张效果转换对象内容
Iris	用特殊形状剪切轮廓扩张或收缩显示效果转换对象内容
Pixelate	这个转换滤镜是一个复杂的视觉效果。在转换的前半段，对象内容先显示为矩形色块拼贴，然后矩形的宽度由一个像素增加至 MaxSquare 属性所设置的值。每个矩形的颜色由其所覆盖区域的像素的颜色平均值决定。接下来转换的后半段，矩形被还原为新内容具体的图像像素，显示出新的内容 在使用此转换滤镜前需要设置此滤镜的 Enabled 特性值为 false。这将预防在转换发生前彩色拼贴效果的静态滤镜先在对象内容上发生作用
RadialWipe	用放射状擦除效果转换对象内容，效果类似汽车挡风玻璃的刮雨刀
RandomBars	用随机发生的线条转换对象内容
RandomDissolve	用随机像素溶解效果转换对象内容
RevealTrans	提供了 24 种转换对象内容的效果
Slide	用滑条抽离效果转换对象内容
Spiral	用矩形螺旋方式转换对象内容
Stretch	用拉伸(缩)变形效果转换对象内容
Strips	用锯齿边覆盖效果转换对象内容
Wheel	用风车叶轮旋转效果转换对象内容
Zigzag	用类似擦地板的效果转换对象内容

以下对上述转换滤镜基本的使用方法进行简单的说明。

1. Spiral 滤镜

该属性用于以矩形螺旋方式转换对象内容，其一般定义方式如下：

```
filter:progid:DXImageTransform.Microsoft.Spiral(enabled=bEnabled, duration=fDuration,
gridSizeX=iColumns, gridSizeY=iColumns)
```

定义中涉及的属性用法如下。

- duration：可选项。浮点数(Real)。设置或检索转换完成所用的时间。其值为秒.毫秒(0.0000)格式；可以使用 play 方法的 iDuration 参数设置转换回放的持续时间。然而，一旦调用了 play 方法，在回放持续过程中 Duration 特性就变为只读特性。
- gridSizeX：可选项，必须为整数值(Integer)，设置或检索滤镜效果中横向盘旋多少次，其取值范围为 1～100，默认值为 16。

- gridSizeY：可选项，必须为整数值(Integer)，设置或检索滤镜效果中纵向盘旋多少次，取值范围为 1～100，默认值为 16。

由于转换滤镜需要在动态情况下发挥作用，因此，使用转换滤镜不仅需要关注滤镜的属性，还需要注意控制好过程中的方法和事件。一般而言，转换滤镜都具有如表 5-9 所示的三种方法和 OnFilterChange 事件，该事件在滤镜发生改变或是滤镜完成时所触发。

表 5-9　转换滤镜的方法

方　法　名	方法的说明
Apply()	将滤镜应用到对象上
Play()	开始转换
Stop()	停止转换

【实例 5-19】Spiral 滤镜的使用

程序代码如 ex5_19.html 所示。

ex5_19.html

```
<HTML>
  <HEAD><TITLE>Spiral 滤镜</TITLE>
    <script language="JavaScript">
      function show()
      {
        wwm.filters.item(0).apply();
        wwm.src="cat0.jpg";
        wwm.filters.item(0).play();
      }
    </script>
  </HEAD>
  <BODY>
    <IMG src="image.jpg" id="wwm" onclick="show()" style="filter:progid: DXImageTransform.
        Microsoft.Spiral(duration=1,gridSizeX=20,gridSizeY=30);width:333; height:222;">
  </BODY>
</HTML>
```

该例中使用了 JavaScript，这是本书第 6 章中将要介绍的知识，由于转换滤镜已经提前使用，读者如果不能理解这部分内容，可以先跳过这部分内容，重点理解有关滤镜的使用方法即可。

运行上面的实例后，首先显示了一幅冰山的图片，在该图片上单击后，出现了在擦除原图时显示出新图片的视觉转换效果。由于选择了 Spiral 滤镜，这里的方式为矩形螺旋状，请读者自行运行该实例，查看实际运行的结果。

使用类似的方法，可以更换其他不同的滤镜来得到更多不同的显示效果。

2. RevealTrans 滤镜

RevealTrans(显示转换)滤镜提供的是一种更为多变的转换效果，它不像只能提供某一

种转换效果的转换滤镜，它同时提供了多达 24 种的效果，其语法为：

```
filter:progid:DXImageTransform.Microsoft.RevealTrans(enabled=bEnabled, duration=fDuration,
transition=iTransitionType)
```

其中的 duration 为转换的秒数，transition 为转换类型代号。表 5-10 列出了所有可用的转换类型和其对应的代号，实际使用中只需指定转换类型的代号，就可以按特有的转换效果进行转换了。

表 5-10　RevealTrans 滤镜的转换效果及其代号说明

显示转换滤镜的转换形式	所对应的代号	显示转换滤镜的转换形式	所对应的代号
矩形从大至小	0	随机溶解	12
矩形从小至大	1	垂直向内裂开	13
圆形从大至小	2	垂直向外裂开	14
圆形从小至大	3	水平向内裂开	15
向上推开	4	水平向外裂开	16
向下推开	5	向左下剥开	17
向右推开	6	向左上剥开	18
向左推开	7	向右下剥开	19
垂直形百叶窗	8	向右上剥开	20
水平形百叶窗	9	随机水平细纹	21
水平棋盘	10	随机垂直细纹	22
垂直棋盘	11	随机选取一种特效	23

【实例 5-20】blendTrans 滤镜的使用

程序代码如 ex5_20.html 所示。

ex5_20.html

```
<HTML>
  <HEAD><TITLE>blendTrans 效果测试</TITLE>
    <SCRIPT language="JavaScript">
      function run()
      {
        show.filters.RevealTrans.apply();
        show.src="cat1.jpg" ;
        show.filters.RevealTrans.play();
      }
    </SCRIPT>
  </HEAD>
  <BODY>
    <IMG ID="show" src="cat0.jpg" onclick=run() style="filter:RevealTrans (duration=3, transition=5);">
  </BODY>
</HTML>
```

在浏览器中运行该例子后，首先显示了一幅一只猫的图片，在该图片上用鼠标单击后，

出现了表 5-10 中定义的第 5 种转换效果——向下推开。请读者自行运行该实例，查看运行的效果。读者还可以进一步更换不同的效果，并查看。

【实例 5-21】blendTrans 滤镜的所有效果展示

程序代码如 ex5_21.html 所示。

ex5_21.html

```
<HTML>
  <HEAD><TITLE>blendTrans 滤镜的所有效果展示</TITLE>
    <SCRIPT language="JavaScript">
      var i = 0;
      function trans()
      {
        obj = document.all["cat"];
        obj.filters.item(0).Transition = i
        obj.filters.item(0).apply();
        obj.src = "cat"+(i%2)+".jpg"
        obj.filters.item(0).play();
        status = "第" + i + "种效果";
        if(i<24) i++;
        else    i=0;
      }
    </SCRIPT>
  <BODY>
    <INPUT TYPE="button" VALUE="转换" onClick="trans()">
    <BR>
    <IMG ID="cat" SRC="cat1.jpg" STYLE="filter:revealtrans(duration=1)">
  </BODY>
</HTML>
```

在浏览器中运行该实例后，读者可以单击“转换”按钮来查看所有的 24 种转换效果，同时在浏览器的状态栏中可显示目前正在显示的是第几种效果，如图 5-17 所示。

注意：

本实例中运用了 JavaScript 技术，读者可在学习了第 6 章之后再次细读本实例。



图 5-17　blendTrans 滤镜的所有效果展示

5.5　关于 CSS3

CSS3 是目前最新的版本，其特点是模块化。从模块的角度看，以前的规范实在是太庞大且较复杂，所以，把它分解为一些小的模块，同时也加入了更多新的模块。这些模块

主要包括盒子模型、列表模块、超链接方式、语言模块、背景和边框、文字特效、多栏布局等。

CSS3 将完全向后兼容，所以没有必要修改早期网站的设计来让它们正常工作。浏览器也还将继续支持 CSS2，CSS3 主要的影响是将可以使用新特性，这将会允许实现新的设计效果(譬如动态和渐变)，而且可以更简单地设计特定的效果(如分栏等)。

5.5.1　页面布局

Web 页面的布局，我们常见的主要有“浮动布局”(float)、“定位布局”(position)、“行内块布局”(inline-block)、“CSS3 的多栏布局”(Columns)、“伸缩布局”(Flexbox)、以及“网格布局”(Grids)等，在众多布局方法中，使用不同的细节能得到不同的布局效果。虽然这些布局能让大家实现常见的布局效果，但在实际中还是存在不少的问题，比如说浏览器的兼容性、修改显示顺序需要调整文档结构等。

CSS3 Grid Layout 是一个新的模块，这个模块主要定义一个二维网格布局系统，用来优化用户界面设计。在这个网格布局模块中，网格容器的所有子元素可以在一个灵活的或者固定的了布局网格中定位到任意槽中。

网格布局可以将应用程序分割成不同的空间，或者定义它们的大小、位置以及层级。就像表格一样，网格布局可以让 Web 设计师根据元素按列或行对齐排列，但与表格不同，网格布局没有内容结构。例如一个网格布局中的子元素都可以定位自己的位置，这样可以重叠和类似元素定位。此外，没有内容结构的网格布局有助于使用流体、调整顺序等技术管理或更改布局。通过结合 CSS 的媒体查询属性，可以控制网格布局容器和它们的子元素，使用页面的布局根据不同的设备和可用空间调整元素的显示风格与定位，而不需要去改变文档结构的本质内容。

【实例 5-22】Grid Layout 展示

程序代码如 ex5_22.html 所示。

ex5_22.html

```
<!DOCTYPE html><html class=''>
<head><meta charset='gb2312'>
<script src='prefixfree.min.js'></script>
<style >#grid {
   display: grid;
   background-color: orange;
   grid-definition-columns: auto minmax(min-content, 1fr);
   grid-definition-rows: auto minmax(min-content, 1fr) auto
}
#title{
   grid-column: 1;
   grid-row: 1 ;
   background-color: red;
}
#score{
```

```
        grid-column: 1;
        grid-row: 3;
        background-color:green;
    }
    #stats{
        grid-column: 1;
        grid-row: 2;
        justify-self: start ;
        background-color:#e9f;
    }
    #board{
        grid-column: 2;
        grid-row: 1 / span 2;
        background-color: #ccc;
    }
    #controls {
        grid-column: 2;
        grid-row: 3;
        align-self: center;
        background-color: yellow;
    }
    </style></head><body>
    <div id="grid">
        <div id="title">游戏标题</div>
        <div id="score">得分</div>
        <div id="stats">状态栏</div>
        <div id="board">面板</div>
        <div id="controls">控制</div>
    </div>
    </body></html>
```

在 head 部分的样式定义的 grid 部分定义了网格。其中将显示分为两列，第一列尺寸由内容大小控制，第二列使用剩余空间，但大小从不会小于游戏主机板和控制区域最小宽度，游戏主面板和游戏控制区域占第二列，共包含三行。第一行和最后一行的大小根据内容决定，中间一行可以使用剩余空间，但从来不会小于游戏主面板的最小高度。游戏的每一部分都是通过网格线来定义的，每个部分都在其占的行中，如果哪个部分占有的行数多于一行，则需要使用跨行来决定。在浏览器中运行该实例的效果如图 5-18 所示。

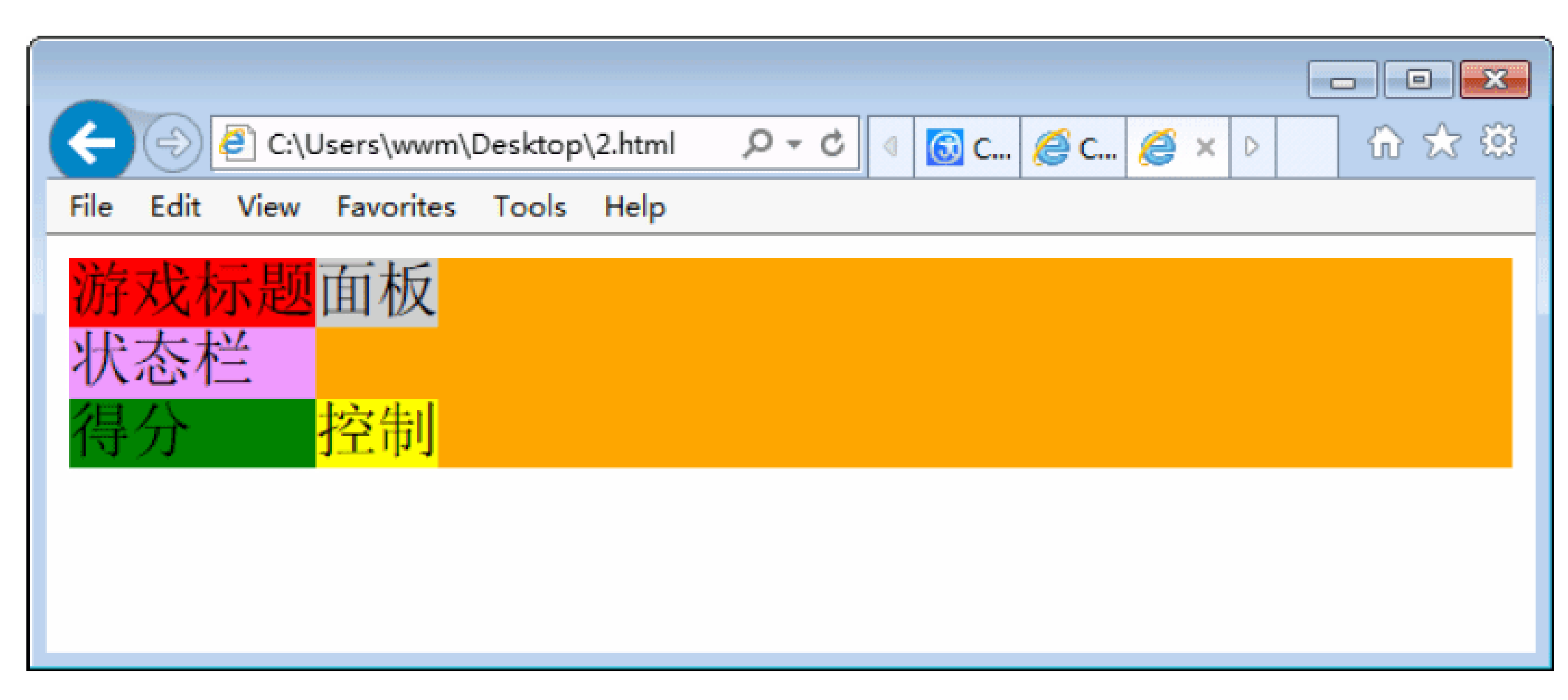


图 5-18　Grid Layout 效果展示

注意：

由于 CSS 3 的很多属性仍然为实验属性，因此使用它们的时候就需要加上各式各样的浏览器前缀。本例中使用了脚本 prefixfree.min.js，它的作用是：帮助自动识别浏览器，生成对应的 CSS 3 样式前缀，这样可以直接当作标准属性来使用了。引用时可以直接通过网站来使用<script src="http://leaverou.github.com/ prefixfree/prefixfree.min.js"></script>。本例使用的方式是访问本地的 prefixfree.min. js 文件。

5.5.2　Flexbox 布局

Flexbox 由伸缩容器和伸缩项目组成。通过设置元素的 display 属性为 flex 或 inline-flex 可以得到一个伸缩容器。设置为 flex 的容器被渲染为一个块级元素，而设置为 inline-flex 的容器则渲染为一个行内元素。

Flexbox 通常能让我们更好地操作它的子元素布局。如果元素容器没有足够的空间，则无须计算每个元素的宽度，就可以设置它们在同一行；可以快速让它们布局在一列；可以方便让它们对齐容器的左、右、中间等；无须修改结构就可以改变它们的显示顺序；如果元素容器设置百分比和视窗大小改变，不用担心未指定元素的确切宽度而破坏布局，因为容器中的每个子元素都可以自动分配容器的宽度或高度的比例。

设置时，需要明确 flex 的值与对应的空间成正比。如果左边栏设置了值为“1”和右边栏设置了值为“2”，伸缩容器剩余的空间将按比例分配给左边栏和右边栏，并且右边栏所占的空间是左边栏的两倍。

以下的实例利用 Flexbox 创建了一个经典的三列布局，主内容宽度为 60%，而边栏是使用“flex”属性，按比例自动根据伸缩容器剩余空间计算得到对应的宽度。

【实例 5-23】Flexbox 用法展示

程序代码如 ex5_23.html 所示。

ex5_23.html

```
<!DOCTYPE html>
<html>
<head>
<title>flexbox 布局</title>
<style type="text/css">
html, body { height: 100%; background: lightgrey; margin: 0;}

.container {
	display: -webkit-flex;
	display: flex;
	-webkit-flex-flow: row;
	flex-flow: row;

	max-width: 1000px;
	height: 100%;
	margin: auto;
```

```
    }
    .main {
        width: 60%;
        margin: 20px 0;
        padding: 7px;
        background: deepskyblue;
    }
    .left {
       -webkit-flex: 1;
       flex: 1;
    }
    .right {
       -webkit-flex: 2;
       flex: 2;
    }
    .nav {
       margin: 20px 15px;
       padding: 7px;

       background: hotpink;
    }
    </style>
    </head>
    <body>

    <div class="container">
    <nav class="nav left">
        <h1>三栏布局左侧</h1>
        <p>使得布局变得非常便利，它不会产生因浮动带来的混乱，也能提供精确的位置控制。</p>
    </nav>

    <section class="main">
        <h1>三栏布局示例</h1>
        <p>Flexbox 使得布局变得非常便利，它不会产生因浮动带来的混乱，也能提供精确的位置控制。
</p>
    </section>

    <nav class="nav right">
        <h1>三栏布局右侧</h1>
        <p>使得布局变得非常便利，它不会产生因浮动带来的混乱，也能提供精确的位置控制。</p>
    </nav>
    </div>
    </body>
    </html>
```

在浏览器中运行该实例的效果如图 5-19 所示，请读者留意代码中有关 flex 的设置及其效果。

值得一提的是，如果采用的浏览器不支持 CSS3，则可能浏览不到正确的结果。图 5-20 所示的显示效果是在 IE 早期版本中浏览的结果。

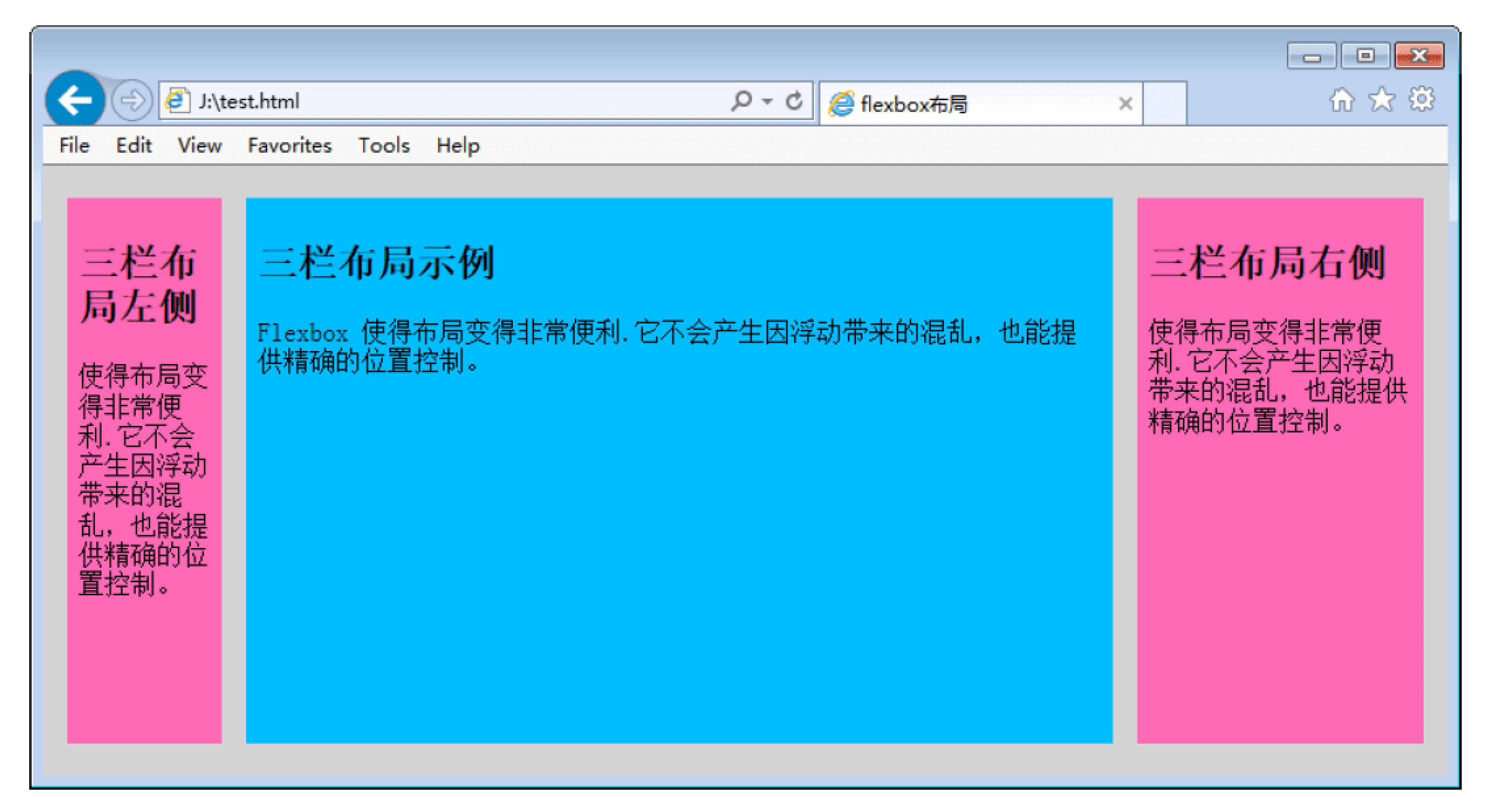


图 5-19　Flexbox 用法展示

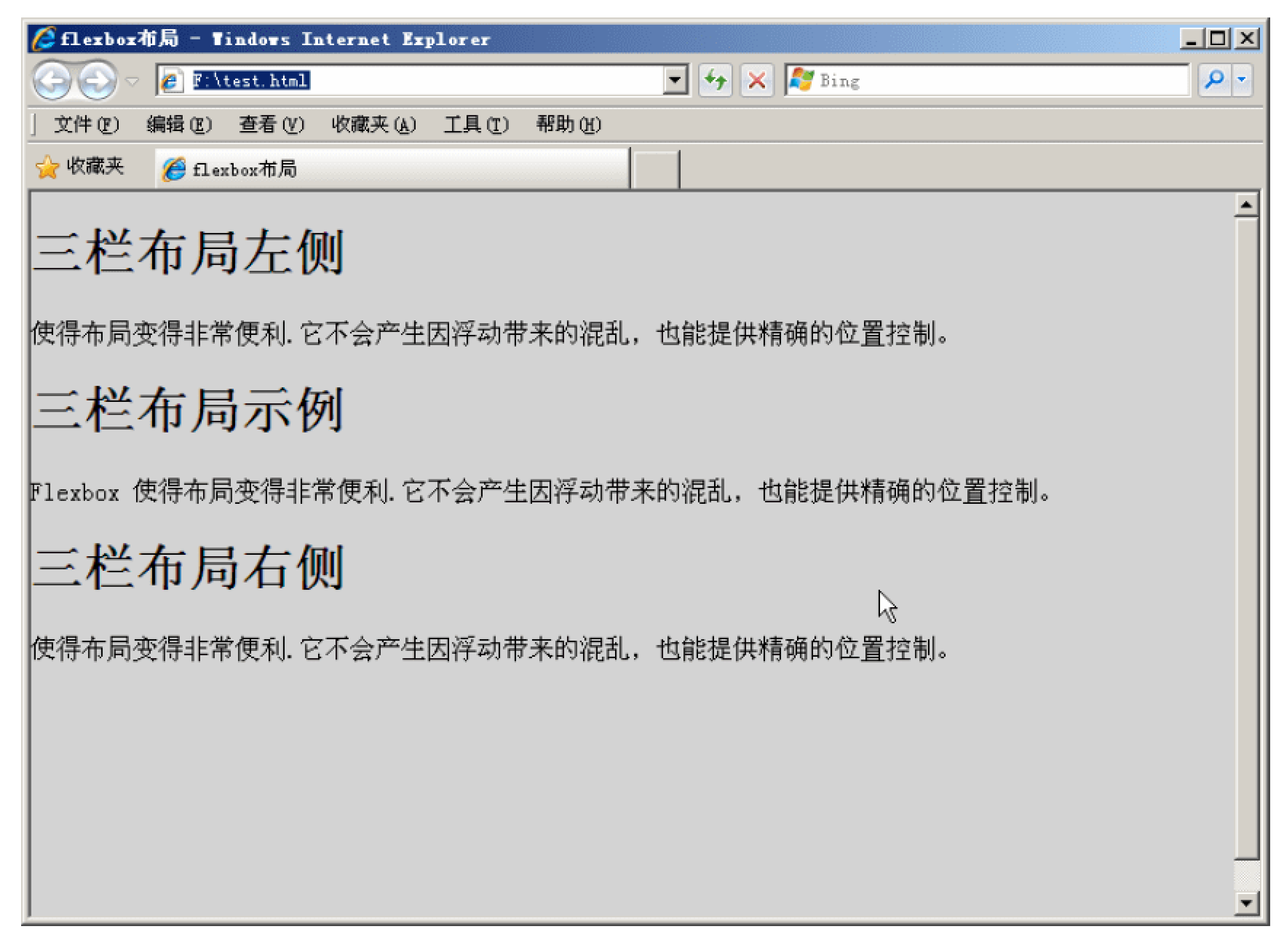


图 5-20　早期 IE 版本使用 Flexbox 的显示效果

5.5.3　边框

CSS 3 提供了关于边框的设置，主要如下。

- border-color：控制边框颜色，并且有了更大的灵活性，可以产生渐变效果；
- border-image：控制边框图像；
- border-corner-image：控制边框边角的图像；
- border-radius：能产生类似圆角矩形的效果。

【实例 5-24】圆角表格用法展示

程序代码如 ex5_24.html 所示。

ex5_24.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

```
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-

transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>CSS 3 圆角表格</title>
<style>
.bordered {
        border: solid #ccc 1px;
        -moz-border-radius: 6px;
        -webkit-border-radius: 6px;
        border-radius: 6px;
        -webkit-box-shadow: 0 1px 1px #ccc;
        -moz-box-shadow: 0 1px 1px #ccc;
        box-shadow: 0 1px 1px #ccc;
}
</style>
</head>
<body>
<table class="bordered">
        <thead>
        <tr>
                <th>#</th>
                <th>IMDB Top 3 Movies</th>
                <th>Year</th>
        </tr>
        </thead>
        <tr>
                <td>1</td>
                <td>The Shawshank Redemption</td>
                <td>1994</td>
        </tr>
        <tr>
                <td>2</td>
                <td>The Godfather</td>
                <td>1972</td>
        </tr>
        <tr>
                <td>3</td>
                <td>The Godfather: Part II</td>
                <td>1974</td>
        </tr>
</table>
</body>
</html>
```

在这个例子中，为了支持多种浏览器，而设置了多个样式，在浏览器中的表格能呈现出圆角效果。图 5-21 所示为浏览器中的显示效果，注意其中所有表格的角显示为圆角。

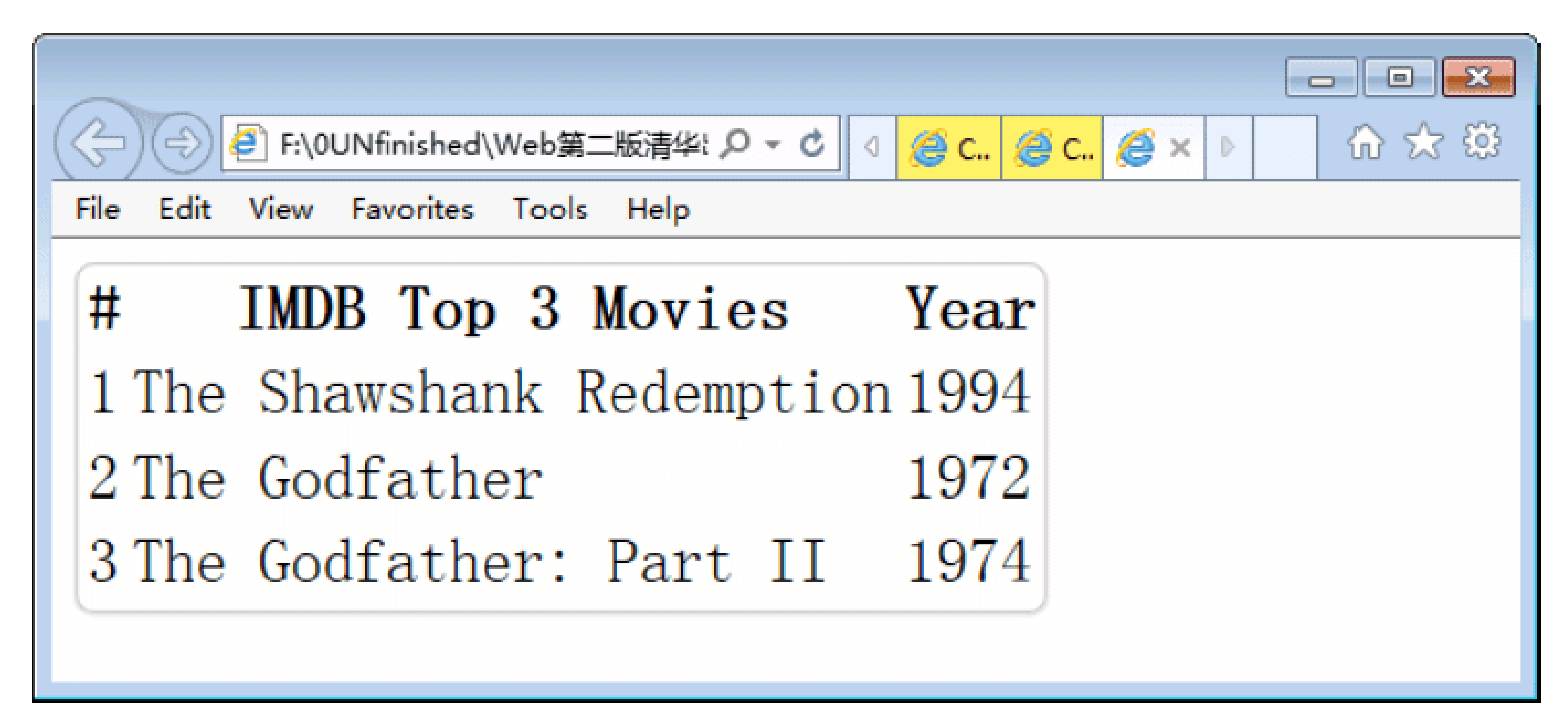


图 5-21　圆角表格

5.5.4　动画

动画包括变形(transform)、转换(transition)和动画(animation)三种类型。

变形包括旋转 rotate、扭曲 skew、缩放 scale 和移动 translate 以及矩阵变形 matrix。

转换主要包含四个属性值，分别是：执行变换的属性(transition-property)、变换延续的时间(transition-duration)、在延续时间段变换的速率变化(transition-timing-function)、变换延迟时间(transition-delay)。

动画中有一项重要的概念“keyframes”，可被称为“关键帧”，使用 transition 制作一个简单的转换效果时，包括了初始属性和最终属性，一个开始执行动作时间和一个延续动作时间以及动作的变换速率。其实这些值都是一个中间值，如果希望控制得更精细，比如用户要第一个时间段执行什么动作，第二个时间段执行什么动作，这样用 transition 就很难实现了，此时也需要使用“关键帧”来控制。那么 CSS 3 的 Animation 就是由“keyframes”这个属性来实现这样的效果的。

keyframes 具有其自己的语法规则，它的命名是由“@keyframes”开头，后面紧接着是这个“动画的名称”加上一对花括号“{}”，括号中就是一些不同时间段样式规则，样式有点像 css。对于一个“@keyframes”中的样式规则是由多个百分比构成的，如“0%”到“100%”之间，可以在这个规则中创建多个百分比，可以分别给每一个百分比中需要有动画效果的元素加上不同的属性，从而让元素达到一种不断变化的效果。比如说移动、改变元素颜色、位置、大小、形状等，不过有一点需要注意的是，可以使用“from”“to”来代表一个动画是从哪开始，到哪结束，也就是说这个“from”就相当于“0%”而“to”相当于“100%”。值得一提的是，其中“0%”不能像别的属性取值一样把百分比符号省略，我们在这里必须加上百分符号(“%”)，如果没有加上的话，这个 keyframes 是无效的，不起任何作用。因为 keyframes 的单位只接受百分比值。

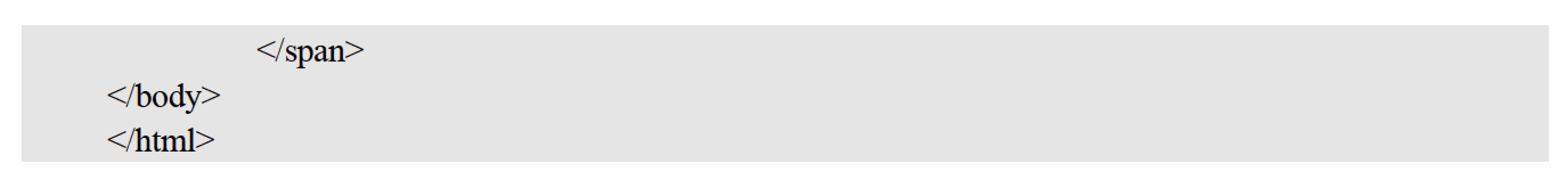
【实例 5-25】动画展示

程序代码如 ex5_25.html 所示。

ex5_25.html

```
<!DOCTYPE html>
<html>
```

```
<head>
<style>
body
{background: #aaccdd;}
@-webkit-keyframes rotate{from{-webkit-transform: rotate(0deg)}to{-webkit-transform: rotate(360deg)}}
@-moz-keyframes rotate{from{-moz-transform: rotate(0deg)}to{-moz-transform: rotate(359deg)}}
@-o-keyframes rotate{from{-o-transform: rotate(0deg)}to{-o-transform: rotate(359deg)}}
@keyframes rotate{from{transform: rotate(0deg)}to{transform: rotate(359deg)}
}
@-webkit-keyframes rotate2{from{-webkit-transform: rotate(0deg)}to{-webkit-transform:
rotate(360deg)}}
@-moz-keyframes rotate2{from{-moz-transform: rotate(0deg)}to{-moz-transform: rotate(359deg)}}
@-o-keyframes rotate2{from{-o-transform: rotate(0deg)}to{-o-transform: rotate(359deg)}}
@keyframes rotate2{from{transform: rotate(0deg)}to{transform: rotate(359deg)}}
.windmill2
{display: block;position: relative;margin: 50px auto;width: 100px;height:120px;}
.windmill2 .pillar
{position: absolute;top: 8px;left: 44px;display: block;height:0;width: 4px;border-width: 0 4px 80px
4px;border-style: none solid solid;border-color:transparent transparent white;}
.windmill2 .axis
{position: absolute;top: 0px;left: 46px;width: 4px;height: 4px;border:3px #fff solid;background: #a5cad6;
border-radius: 5px;z-index: 88;-webkit-transition-property: -webkit-transform;-webkit-transition-duration: 1s;
-moz-transition-property: -moz-transform;-moz-transition-duration: 1s;-webkit-animation: rotate 4s linear infinite;
-moz-animation: rotate 4s linear infinite;-o-animation: rotate 4s linear infinite;animation: rotate 4s linear infinite;}
.windmill2 .swing
{position: absolute;top: 1px;left: -2px;display: block;height: 0;width:2px;border-width: 50px 2px 0px 2px;
border-style: solid solid none;border-color: white transparent transparent ;box-shadow: 1px 1px 1px rgba(105, 97,
97, 0.1);-webkit-transform-origin: 0px 0px;-webkit-transform: rotate(60deg);-moz-transform-origin: 0px 0px;
-moz-transform: rotate(60deg);-ms-transform-origin: 0px 0px;-ms-transform: rotate(60deg);-o-transform-origin:
0px 0px;-o-transform: rotate(60deg);transform-origin: 0px 0px;transform: rotate(60deg);}
.windmill2 .swing2
{position: absolute;top: 0px;left: 4.5px;display: block;height: 0;width: 2px;border-width: 50px 2px 0px
2px;border-style: solid solid none;border-color: white transparent transparent ;-webkit-transform-origin: 0px
0px;-webkit-transform: rotate(180deg);-moz-transform-origin: 0px 0px;-moz-transform: rotate(180deg);
-ms-transform-origin: 0px 0px;-ms-transform: rotate(180deg);-o-transform-origin: 0px 0px;-o-transform:
rotate(180deg);transform-origin: 0px 0px;transform: rotate(180deg);}
.windmill2 .swing3
{position: absolute;top: 6px;left: 3px;display: block;height: 0;width:2px;border-width: 50px 2px 0px 2px;
border-style: solid solid none;border-color: white transparent transparent ;-webkit-transform-origin: 0px 0px;
-webkit-transform: rotate(300deg);-moz-transform-origin: 0px 0px;-moz-transform: rotate(300deg);
-ms-transform-origin: 0px 0px;-ms-transform: rotate(300deg);-o-transform-origin: 0px 0px;-o-transform:
rotate(300deg);transform-origin: 0px 0px;transform: rotate(300deg);}
</style>
</head>
<body>
    <span class="windmill2">
    <span class="pillar"></span>
    <span class="axis">
    <span class="swing"></span>
    <span class="swing2"></span>
    <span class="swing3"></span>
    </span>
```



```
            </span>
    </body>
    </html>
```

在这个例子中，为了支持多种浏览器，而设置了多个样式，在浏览器中显示出一个转动的风车，图 5-22 所示为浏览器中的显示效果，读者可留意源码中动画的实现部分。

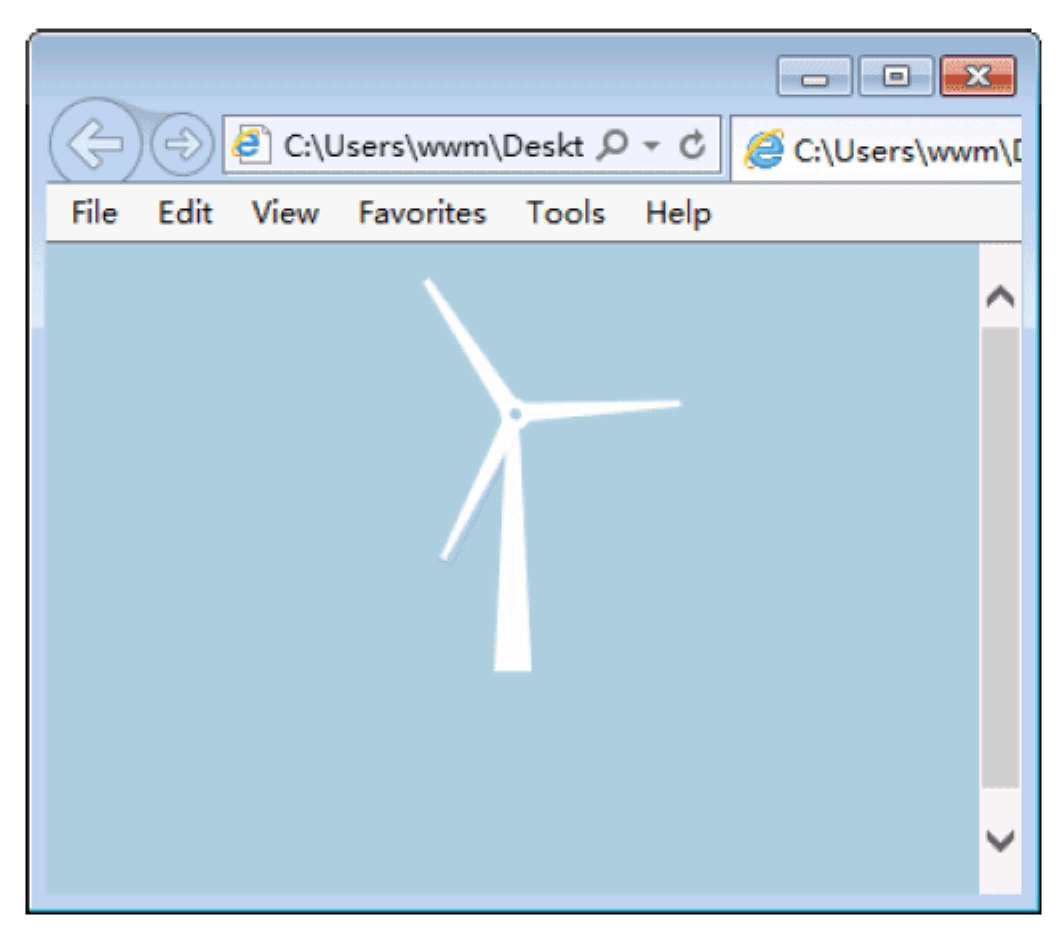


图 5-22　CSS 3 动画效果

5.5.5　选择器

CSS3 增加了更多的 CSS 选择器，可以实现更简单但是更强大的功能，如 nth-child()等。

一些选择器的功能介绍如下。

- Attribute selectors：在属性中可以加入通配符，包括^,$,*等。
- [att^=val]：表示开始字符是 val 的 att 属性。
- CSS3 选择器[att$=val]：表示结束字符是 val 的 att 属性。
- [att*=val]：表示包含至少有一个 val 的 att 属性。

由于 CSS3 中部分功能实现代码较为复杂，因此选用合适的选择器编辑工具可以更方便地实现 CSS 的编辑功能，以下的几个编辑器都能很好地支持 CSS 3 功能，读者可以根据自己的需要和习惯进行灵活的选择。

- CSS3 Maker：CSS 3 Maker 的功能非常强大，可在线编辑并实时演示渐变、阴影、旋转、动画等非常多的效果，并生成对应效果的代码。
- CSS3 Generator：非常不错的各种 CSS3 代码生成器，支持圆角、渐变、旋转和阴影等众多新特性，且带有直接预览效果的功能。
- CSS3 Please：一款实用的 CSS3 工具，可即时在线修改代码并预览效果，还提供了详细的浏览器兼容情况。

5.6　CSS 典型用法实例

5.6.1　使用滤镜制作文字特效

滤镜不仅可以用于制作图片特效，也可以用于生成文字特效，【实例 5-26】使用了 blendtrans 滤镜来生成文字淡入淡出的特效。

【实例 5-26】使用滤镜制作文字特效

程序代码如 ex5_26.html 所示。

ex5_26.html

```
<HTML>
  <HEAD><TITLE>使用滤镜来制作文字特效</TITLE>
    <SCRIPT language="JavaScript">
      function show()
      {
                  if (wwm.filters(0).status==2){
                      wwm.filters(0).stop();
                      if (wwm.style.visibility=="hidden")
                           wwm.style.visibility="visible";
                      else
                         wwm.style.visibility="hidden";
                      window.settimeout("togglemultimedia()",1);
                }
                wwm.filters(0).apply();
                if (wwm.style.visibility=="hidden")
                     wwm.style.visibility="visible";
                 else
                   wwm.style.visibility="hidden";
                   wwm.filters(0).play();
      }
    </SCRIPT>
  </HEAD>
  <BODY>
    <SPAN id="text1" onclick="show()">请点击这里查看效果</span>
    <DIV id="wwm" style="width:100%;color:blue;filter:blendtrans(duration=5)">
      <SPAN id="text1">这段蓝色文字加上了逐渐淡出的多媒体效果</SPAN>
    </DIV>
  </BODY>
</HTML>
```

本例的 blendtrans 滤镜用于<DIV>标签，通过设置该标签的 id 属性，使得<HEAD>部分的 JavaScript 代码可以访问这个文本元素，并将滤镜应用于该文本元素。此外，本例还使用了在普通文本上定义事件来捕捉用户操作事件的方法，本例的“请点击这里查看效果”这段文字看起来是普通文本，但因为定义了 onclick="show()"的代码，因此可以在用户单击时调用 show()函数。在浏览器中运行时，读者可以单击“请点击这里查看效果”来观察下面的文字所呈现的淡入淡出转换效果，如图 5-23 所示。

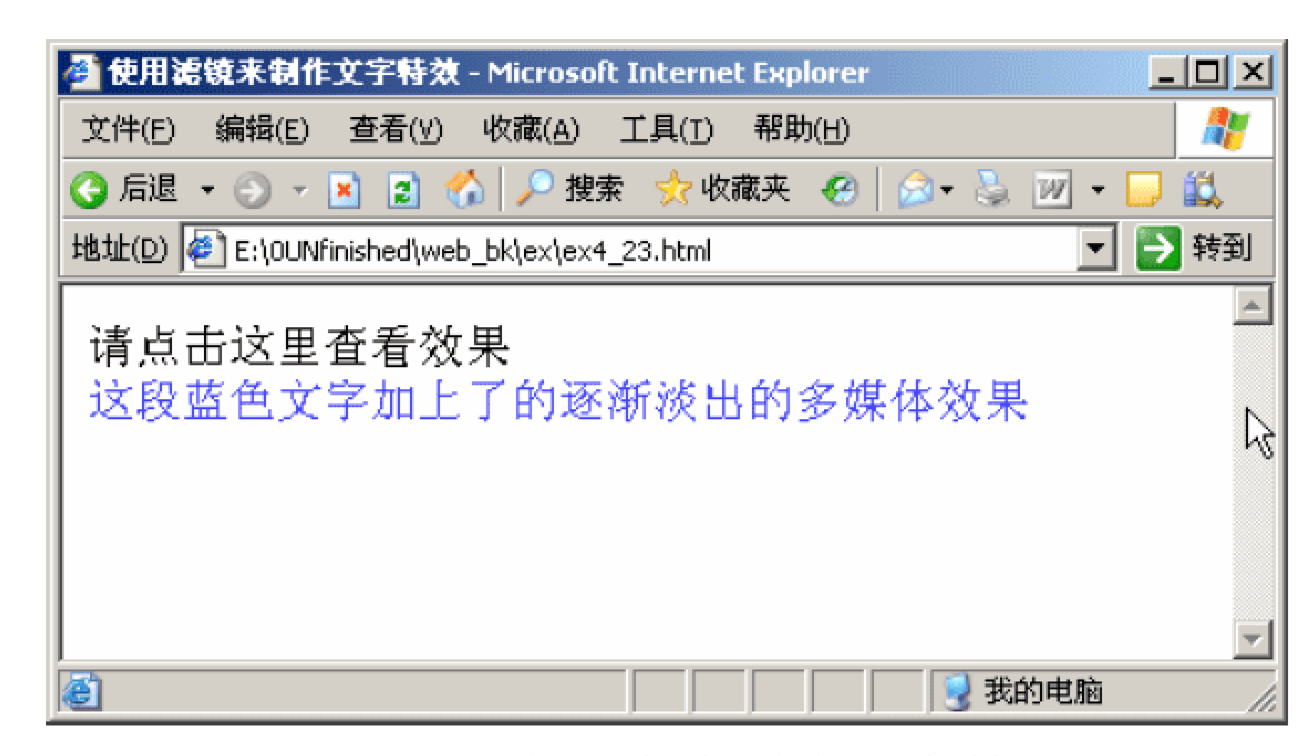


图 5-23　使用滤镜制作文字特效

5.6.2　使用 CSS 来改变浏览器的默认显示样式

一般而言，浏览器的默认样式是可以接受的，但有时为了配合网页风格的需要，希望改变浏览器的默认显示风格，此时，CSS 就能是一个利器。

【实例 5-27】改变滚动条的样式

程序代码如 ex5_27.html 所示。

ex5_27.html

```
<HTML>
  <HEAD><TITLE>改变浏览器的滚动条</TITLE>
    <STYLE>
      html,body{
        scrollbar-face-color: #99ccff;
        scrollbar-shadow-color: #ccccff;
        scrollbar-highlight-color: #ccccff;
        scrollbar-3dlight-color: #99ccff;
        scrollbar-darkshadow-color: #ccccff;
        scrollbar-track-color: #ccccff;
        scrollbar-arrow-color: #000033;
      }
      .highlight{
        scrollbar-face-color: #99ccff;
        scrollbar-shadow-color: #ccccff;
        scrollbar-highlight-color: #ccccff;
        scrollbar-3dlight-color: #99ccff;
        scrollbar-darkshadow-color: #ccccff;
        scrollbar-track-color: #ccccff;
        scrollbar-arrow-color: #000033;
      }
    </STYLE>
  </HEAD>
  <BODY>
    <FORM>
      <TEXTAREA class="highlight"></TEXTAREA>
    </FORM>
  </BODY>
</HTML>
```

本例通过样式表修改了滚动条的样式，在<STYLE>标签中，首先通过选择符组为<HTML>和<BODY>赋予新的滚动条定义，之后又使用同样的方式定义了 highlight 选择符，在<BODY>的<TEXTAREA>中使用了上面所定义的 highlight 选择符，最终的显示效果如图 5-24 所示。

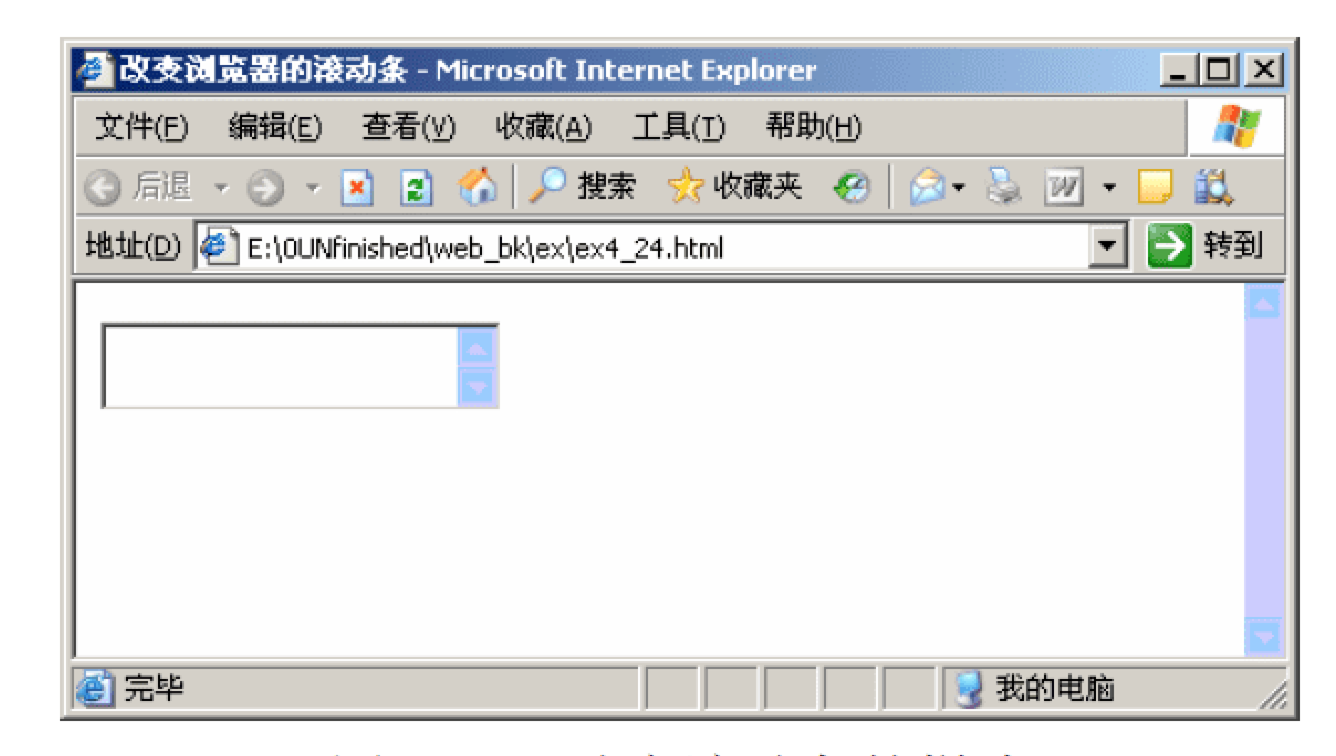


图 5-24　改变滚动条的样式

5.6.3　制作可交互的 360 度全景展示

在商品展示中经常希望制作具有 360 度全景交互的物品展示，其实在网页中也可以通过 CSS 3 技术来实现这种效果，【实例 5-28】给出了一个简单的实现方法。

【实例 5-28】制作可交互旋转的广告展示

程序代码如 ex5_28.html 所示。

ex5_28.html

```
<!DOCTYPE HTML>
  <HEAD>
    <STYLE type="text/css">
     #coke
     {     width: 510px;
           height: 400px;
           margin: 0 auto;
           overflow: auto;
     }
     img
     {     border: 0;
           margin-left: -172px;
     }
     a
     {     display: block;
           padding-top: 19px;
           width: 194px;
     }
     div div
     {     background-image: url(images/coke-scroll.png);
           background-repeat: no-repeat;
           background-position: 0 0;
           padding-left: 300px;
           width: 660px;
     }
     p
     {     margin: 0;
           padding: 0;
           float: left;
           height: 336px;
           background-image: url(images/coke-label.jpg);
           background-attachment: fixed;
           background-repeat: repeat-x;
           width: 1px;
     }
     #x1 {background-position: 5px 30px;}     #x2 {background-position: 0px 30px;}
     #x3 {background-position: -3px 30px;}    #x4 {background-position: -6px 30px;}
     #x5 {background-position: -8px 30px;}    #x6 {background-position: -10px 30px;}
     #x7 {background-position: -12px 30px;}   #x8 {background-position: -14px 30px;}
     #x9 {background-position: -15px 30px;}   #x10 {background-position: -16px 30px;}
     #x11 {background-position: -17px 30px;}  #x12 {background-position: -18px 30px;}
```

```
        #x13 {background-position: -19px 30px;}  #x14 {background-position: -20px 30px;}
        #x15 {background-position: -21px 30px;}  #x16 {background-position: -22px 30px; width: 2px;}
        #x17 {background-position: -23px 30px;}  #x18 {background-position: -24px 30px; width: 2px;}
        #x19 {background-position: -25px 30px; width: 2px;}
        #x20 {background-position: -26px 30px; width: 2px;}
        #x21 {background-position: -27px 30px; width: 2px;}
        #x22 {background-position: -28px 30px; width: 3px;}
        #x23 {background-position: -29px 30px; width: 3px;}
        #x24 {background-position: -30px 30px; width: 4px;}
        #x25 {background-position: -31px 30px; width: 5px;}
        #x26 {background-position: -32px 30px; width: 7px;}
        #x27 {background-position: -33px 30px; width: 12px;}
        #x28 {background-position: -34px 30px; width: 55px;}
        #x29 {background-position: -35px 30px; width: 11px;}
        #x30 {background-position: -36px 30px; width: 6px;}
        #x31 {background-position: -37px 30px; width: 5px;}
        #x32 {background-position: -38px 30px; width: 4px;}
        #x33 {background-position: -39px 30px; width: 3px;}
        #x34 {background-position: -40px 30px; width: 2px;}
        #x35 {background-position: -41px 30px; width: 3px;}
        #x36 {background-position: -42px 30px; width: 2px;}
        #x37 {background-position: -43px 30px; width: 2px;}
        #x38 {background-position: -44px 30px;}  #x39 {background-position: -45px 30px; width: 2px;}
        #x40 {background-position: -46px 30px;}    #x41 {background-position: -47px 30px;}
        #x42 {background-position: -48px 30px;}    #x43 {background-position: -49px 30px;}
        #x44 {background-position: -50px 30px;}    #x45 {background-position: -51px 30px;}
        #x46 {background-position: -52px 30px;}    #x47 {background-position: -53px 30px;}
        #x48 {background-position: -54px 30px;}    #x49 {background-position: -56px 30px;}
        #x50 {background-position: -58px 30px;}    #x51 {background-position: -60px 30px;}
        #x52 {background-position: -62px 30px;}    #x53 {background-position: -65px 30px;}
        #x54 {background-position: -68px 30px;}    #x55 {background-position: -74px 30px;}
    </STYLE>
      </HEAD>
      <BODY>
        <DIV id="coke">
          <DIV id="y">
            <P id="x1"></P>          <P id="x2"></P>          <P id="x3"></P>
            <P id="x4"></P>          <P id="x5"></P>          <P id="x6"></P>
            <P id="x7"></P>          <P id="x8"></P>          <P id="x9"></P>
            <P id="x10"></P>         <P id="x11"></P>         <P id="x12"></P>
            <P id="x13"></P>         <P id="x14"></P>         <P id="x15"></P>
            <P id="x16"></P>         <P id="x17"></P>         <P id="x18"></P>
            <P id="x19"></P>         <P id="x20"></P>         <P id="x21"></P>
            <P id="x22"></P>         <P id="x23"></P>         <P id="x24"></P>
            <P id="x25"></P>         <P id="x26"></P>         <P id="x27"></P>
            <P id="x28"></P>         <P id="x29"></P>         <P id="x30"></P>
            <P id="x31"></P>         <P id="x32"></P>         <P id="x33"></P>
            <P id="x34"></P>         <P id="x35"></P>         <P id="x36"></P>
            <P id="x37"></P>         <P id="x38"></P>         <P id="x39"></P>
            <P id="x40"></P>         <P id="x41"></P>         <P id="x42"></P>
            <P id="x43"></P>         <P id="x44"></P>         <P id="x45"></P>
            <P id="x46"></P>         <P id="x47"></P>         <P id="x48"></P>
```

```
            <P id="x49"></P>            <P id="x50"></P>            <P id="x51"></P>
            <P id="x52"></P>            <P id="x53"></P>            <P id="x54"></P>
            <P id="x55"></P>
            <a href="http://www.njupt.edu.cn"><IMG src="images/coke-can.png"></A>
        </DIV>
    </DIV>
  </BODY>
</HTML>
```

本例通过 P 标签实现了滚动条，再利用样式表中的 background-position 属性设置了多个显示位移的不同位置，只要设置的角度足够多，就可以生成连续的转动效果。浏览器中的显示效果如图 5-25 所示。读者需要在浏览器中用鼠标拖动进度条来查看本例的实际运行效果。

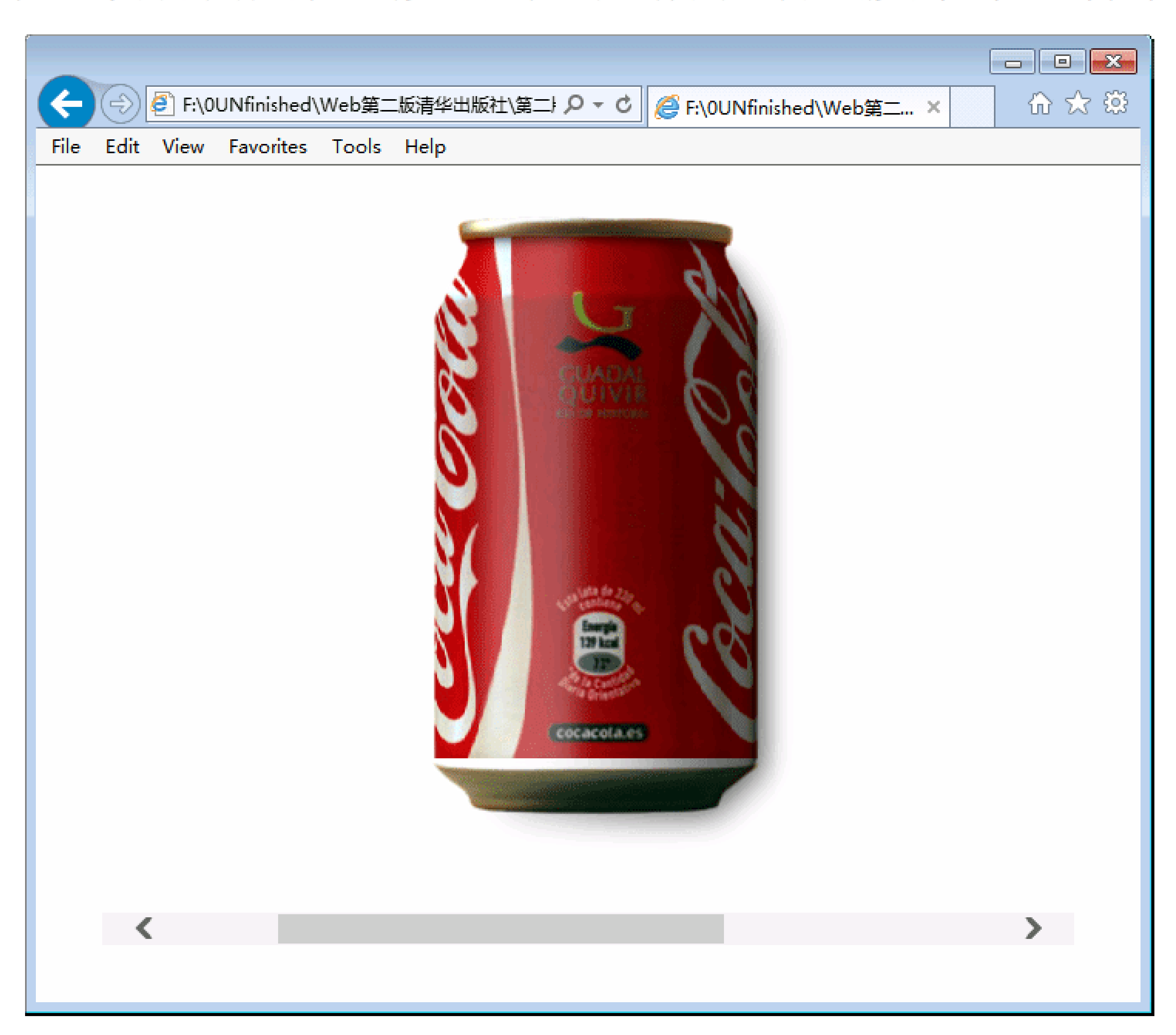


图 5-25　制作可交互旋转的广告效果

5.6.4　自动适应移动设备横竖屏显示方式的实现方案

如果希望制作一个能在手机上进行浏览的网站，除了需要考虑分辨率外，还需要考虑手机存在横竖屏的不同显示方式，这可以利用 CSS3 的@media orientation 功能来自动匹配。

@media 是在 CSS3 中定义的，功能非常强大，由于桌面电脑一般是无法出现横竖屏两种显示方式的，所以 orientation 通常只对移动设备起作用。

头部声明代码如下:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0,user-scalable=no,
maximum-scale=1.0" >
```

利用 media 匹配屏幕是横屏还是竖屏的代码如下：

```
@media all and (orientation:landscape) {   /*横屏的状态，横屏时的 css 代码*/
   body {
      background-color: #880000;
   }
}
@media all and (orientation:portrait){     /*匹配竖屏的状态，竖屏时的 css 代码*/
```

```
    body {
        background-color:#008800;
    }
}
```

实际应用中需要考虑的几个问题包括如下。

- 手机 Web 页面元素内容一般都是通过百分比定义的，以便在不同分辨率的屏幕上都能正常显示。但是由于移动设备的屏幕分辨率差异很大，同样的页面在屏幕翻转过来时可能使得利用百分比方式定义的元素会变得非常大，影响美观性。因此，如果用 orientation 匹配屏幕的翻转状态，就可以编写不同的 CSS 程序代码加以控制页面样式。
- 对于有背景图的移动 Web 页面，可以根据 orientation 匹配屏幕状态，设置不同的背景图。
- 对于采用绝对位置来定位某些元素的 Web 页面，将某元素定位到页面底部，当屏幕是竖屏状态时，可能因为页面总长度小于屏幕高度(但是大于屏幕宽度)，这时将绝对定位元素定位到底部是正确的。但是当屏幕翻转成为横屏时，此时因为页面内容高度大于屏幕高度(就是未翻转时屏幕宽度)，绝对定位元素会覆盖在页面内容之上，导致页面出现问题，这时可用 orientation 匹配屏幕状态，调整 CSS 代码。
- 关于匹配屏幕横竖屏状态还可通过 JavaScript 判断，在 JavaScript 中可以通过监听 onorientationchange 事件来实现屏幕横竖屏之间的切换。

5.7　本章小结

掌握了 CSS 技术，可以对 HTML 文档的布局、字体、颜色、背景和其他图文效果实施统一、精确的控制，增强网页的外观效果。此外，还能够简化网页的格式代码，加快下载和显示的速度，减少网页代码量，减轻网页建设时重复劳动的工作量。

本章首先介绍了什么是层叠式样式表(CSS)技术。其次，具体说明了使用 CSS 控制样式的方法。再次，对于 CSS 应用的另一个重要方面 CSS 滤镜进行了较为详细的说明。最后，介绍了 CSS3 中的基本用法及一些典型的 CSS 制作案例。

5.8　思考和练习

1. 为网页添加样式表有四种不同的方式，在应用时该如何加以灵活选择？
2. 给整个网页增加背景色很容易做到，但如果给一部分文字加背景色该如何定义呢？
3. 在定义动态链接时，页面中所有的链接效果都会改变，如果想在一个页面中定义两组以上的链接效果，怎样定义？
4. 如何给某部分网页元素添加边框？

第6章　JavaScript语言

JavaScript 是一种脚本语言，为网页设计提供了一种在客户端运行程序的手段，通过它可以实现客户端数据验证、网页特效等功能。本章讲述 JavaScript 的基础知识，介绍 JavaScript 在客户端运行的工作机制，在此基础上，读者可以进一步掌握 JavaScript 的对象化编程的方法和各种常用对象的使用。

本章要点：

- JavaScript 脚本语言的基本概念、基本语法
- 在网页中插入脚本语言的方式
- JavaScript 的变量、各类控制语句和函数的用法
- JavaScript 内置对象和文档对象模型的基本用法

6.1　JavaScript 简介

在浏览网页时，当鼠标单击或移过某些网页元素时，会产生某些特别的效果，如欢迎辞、警示语或者状态栏出现提示等。有些网页在提交数据时需要较长的时间才能得到响应，而有些网页却可以得到立即响应，这是由于不同网页根据其自身需要采用了不同的技术实现方案而导致的，多种技术方案中一种非常重要的方案就是 JavaScript。

JavaScript 可以增加网页的互动性；使 HTML 代码中重复的代码得以简化，减少网页载入时间；因为无须将数据发送到服务器，它也能即时响应用户的操作，对提交的表单做即时检查等，因此某种程度上提升了用户的体验。可以说对 JavaScript 的利用是无穷无尽的，受到限制的只有创意。

6.1.1　什么是 JavaScript

JavaScript 最初是由 Netscape 公司的 Brendan Eich 发明的，当时被称为 LiveScript。1995 年 Java 出现后，在 LiveScript 中引入了 Java 的部分设计理念，还增加了对 Java Applet 的支持，同时将其改名为 JavaScript，于 1996 年 2 月随 Netscape Navigator 2.0 正式推出了 JavaScript 1.0 版本。Microsoft 公司也在 IE 3.0 中支持了与 JavaScript 1.2 兼容的 JScript，并在其后来的版本中进行了一定的扩充。

网景公司与微软公司分别将各自的脚本语言交给欧洲计算机制造商联合会(European Computer Manufacturers Association，ECMA)。ECMA 于 1997 年 6 月公布了 Web 脚本语言标准 ECMA-262(ECMAScript Language Specification)。ECMA 将该标准提交给国际标准化

组织，经过少量修改后，1998 年 4 月它成为国际标准：ISO/IEC 16262(Information technology - ECMAScript language specification)。ECMA 于 1998 年 6 月推出了与 16262 国际标准完全兼容的第二版(2rd Edition)：ECMA 262-2，1999 年 12 月又推出了第三版(3rd Edition)：ECMA 262-3。ISO 于 2002 年 6 月 13 日又推出了 16262 的第二版：ISO/IEC 16262:2002。

JavaScript 是一种基于对象(Object)和事件驱动(Event Driven)，并具有较高安全性能的脚本语言。它能与 HTML 超文本标记语言、Java 脚本语言(Java 小程序)等技术融合在一起，实现在一个 Web 页面中连接和控制多个对象，实现与 Web 客户交互的功能。利用它所开发的客户端应用程序，是嵌入在标准的 HTML 语言中的，弥补了 HTML 语言本身的缺陷，它具有以下几个特点。

1. 是一种脚本语言

JavaScript 采用小程序段的编程方式，直接将代码写入 HTML 文档，当浏览器下载并读取时才进行编译，随后执行。所以查看 HTML 源文件就能查看到其中嵌入的 JavaScript 源代码。这种解释性的编程语言，提供了一个方便、简单的开发过程。它的语法和结构与 C、C++、C#、VB、Delphi、Java 等编程语言十分类似，不同之处在于其运行时不会出现独立的运行窗口，运行结果是借助浏览器来展示的。

在 HTML 文档中，JavaScript 是采用<Script>...</Script>标签嵌入网页中的。

注意：

部分读者看到 Java 和 JavaScript 都有“Java”，会倾向于认为它们类似。其实它们差异非常大，JavaScript 也不是 Java 的精简版。

2. 基于对象

JavaScript 是一种基于对象但不是完全面向对象的语言。之所以说它基于对象，主要是因为它没有提供如抽象、继承、重载等有关面向对象语言的许多功能。而是把其他语言所创建的复杂对象统一起来，用系统中的类来创建对象，也可以自己创建类来生成对象，从而形成一个较为强大的对象系统。使用它能编写出具有一定可复用性和封装性的代码。

3. 简单性

总的来说，它不具备完全面向对象语言的全部复杂功能，因而较为简单。具体来说，首先它采用了基于 Java 基本语句和控制流，并在此基础上进行了一定的简化，是一种结构紧凑的语言；其次是它的变量类型采用弱类型，不采用严格的数据类型从而简化了编程；此外还有一些其他方面的简化。

比如 Java 编程语言采用了强类型变量检查，即所有变量在编译之前必须作声明，代码如下：

```
Integer x;
String y;
x=3;
```

```
y=”1357”;
```

其中 x 是一个整数，y 是一个字符串。

JavaScript 中变量声明采用弱类型，即变量在使用前不需作声明，而是解释器在运行时再检查其数据类型，代码如下：

```
x=3;
y="1357";
```

前者说明 x 为数值型变量，而后者说明 y 为字符型变量，在前面可以没有专门的定义语句。

4. 安全性

JavaScript 是一种较为安全的语言，它不允许访问本地磁盘，也不能将数据保存在服务器上，不允许对网络文档进行修改和删除，只能通过浏览器实现信息浏览或动态交互，这些限制有效地避免了一些可能出现的不安全操作。

5. 动态性

JavaScript 是动态的，它可以直接使用本机的运算资源对用户或客户的输入做出快速响应，此过程无须使用网络调用服务器端程序。其对客户产生的响应是采用以事件驱动的方式进行的，用户按下鼠标、移动窗口、选择菜单等均构成事件并可能触发事件。通过事件处理程序可以执行相应的代码功能，因此其具有快速的动态响应特征。

6. 跨平台性

JavaScript 依赖于浏览器，与操作系统及硬件环境无关，只要计算机安装了支持 JavaScript 的浏览器，就可以正确执行。一定程度上实现了“编写一次，到处运行”的梦想。

综上所述，JavaScript 是一种描述语言，采用解释执行方式，是一种基于对象的脚本语言。尽管与 Java 这类完全面向对象的语言相比，它的功能要弱一些，但对于运行环境和需求而言，已经足够了。

当然，JavaScript 也存在一些缺点，如各种浏览器对 JavaScript 的支持程度存在差异，相同的 JavaScript 脚本在不同浏览器中的运行效果不尽相同；为了提高安全性，牺牲了一部分功能等。

6.1.2　作用

对于网站开发，JavaScript 具有以下作用。

1. 创建生动的用户界面

为了使页面更加生动活泼，常需要在按钮被按下时作出某些特殊的响应，对此首先可以考虑使用表单中的普通按钮，虽然这样也能实现所需功能，但却往往会破坏整个页面的和谐与美观；当然，也可以采用超链接来实现同样的功能，但这样就只能实现按钮被按下

的效果，而不能展现按钮从按下到弹起的整个过程。在这种情况下，如果能合理运用 JavaScript，就可以利用图片或动画来实现该功能：制作两幅图片，分别是按钮正常状态及被按下的图片；开始时在页面上放置正常状态的图片，当鼠标在图片的范围内单击时，图片得以切换，鼠标释放时再恢复为正常状态的图片，利用这个原理就能既保持页面的美观，又使页面变得生动活泼。当然在用户界面方面，JavaScript 还有更强大的功能。

2. 数据有效性验证工作

当用户填写表单并提交表单数据时，可能因为用户的疏忽而有所遗漏，或者由于用户的玩笑而填入了无效数据，对此该如何处理呢？虽然表单数据可送至服务器上的处理程序(如 PHP、ASP、JSP、.net 等)进行处理，但即使所输入的数据只有一项不合格，用户也必须等待一个完整的 Web 交互周期之后才能看到反馈的结果，况且系统管理员也不希望那种毫无意义的玩笑数据加重服务器的处理负担。为了达到上述的目的，运用 JavaScript 使得数据的有效性验证在客户端进行，对于遗漏数据以及无效数据等，在表单提交到服务器之前得到检验并立即反馈给用户，避免了无效的网络传输。

3. 数据查找

现在许多网页中都包含了搜索功能。用户填入关键词，浏览器将其发送给服务器，服务器启动数据库搜索引擎，最后将检索结果反馈给用户。这种方案虽没有什么不妥，但在数据量不大的情况下，如只是在几十条数据中进行检索，虽然它也能够实现数据查找的功能，但与用户的等待时间、服务器加重的负载相比，采用这种方案有些得不偿失。既然数据量不大，也可以根据需要采用 JavaScript 来检索已存放在客户端数据的解决方案。

注意：

使用了 JavaScript 后，需要在常用的几种浏览器中进行严格的测试，来确保兼容性。以往曾出现过因为使用 JavaScript 后在不同浏览器中出现不同显示效果的现象。当然，如果开发时用 IE 进行了测试，若能保证所有用户仅使用 IE，则该项测试可以忽略。

6.1.3　JavaScript 语言的组成

在 JavaScript 语言中分为 3 个部分：JavaScript 核心语言、JavaScript 客户端扩展、JavaScript 服务器端扩展，下面对这 3 个部分进行简要介绍。

1. JavaScript 核心语言

JavaScript 内置的对象——Array 对象、Date 对象和 Math 对象等，JavaScript 核心语言中定义的是在客户端和服务器端都会用到的基本语法。

2. JavaScript 客户端扩展

在客户端运行的 JavaScript 在核心语言的基础上扩展了控制浏览器和文本的对象模型 DOM(Document Object Model)。将 JavaScript 核心语言部分和 JavaScript 客户端扩展结合起

来，通过脚本对页面上的对象进行控制，完成诸如在页面中处理鼠标单击、表单输入以及控制页面的浏览等功能。由于对于 JavaScript 客户端扩展部分的标准化还不是很完善，因此不同浏览器对客户端 JavaScript 的支持还存在一定差异。

3. JavaScript 服务器端扩展

在服务器端运行的 JavaScript 在核心语言的基础上扩展了在服务器上运行时需要的对象，这些对象既可以与数据库互连，也可以对服务器上的文件进行控制，还可以在应用程序之间交换信息。服务器端运行的 JavaScript 应用将 JavaScript 核心语言部分和 JavaScript 服务器端扩展结合起来，在服务器上编写的脚本中，可以实现同样的功能，不同之处在于 JavaScript 服务器端扩展仍然和 HTML 页面结合在一起，因而和其他技术相比，更加易于开发和维护。热门的 AJAX 就是 JavaScript 在服务器端扩展的一个实例。

6.1.4 JavaScript 引入网页的方式

JavaScript 需要嵌入在网页中才能使用，这里介绍将 JavaScript 嵌入网页的方法。

1. 直接将 JavaScript 嵌入网页

JavaScript 可出现在网页中的任意位置，但必须使用标记<script>…</script>。如果希望在声明框架的网页(框架网页)中使用，则一定要在<frameset>之前插入，否则这些代码可能不会被运行。JavaScript 的基本格式如下：

```
<script>
<!--  // 第 2 行
   ...
   (JavaScript 代码)
   ...
//-->  //第 6 行
</script>
```

第 2 行和第 6 行的作用，是让不能解析<script>标签的浏览器忽略 JavaScript 代码。一般可以省略，因为现在绝大多数浏览器能支持。第 6 行前边的双反斜杠“//”是 JavaScript 里的注释标号。

注意：

通常将JavaScript代码放置于<HEAD></HEAD>或者<BODY></BODY>之间。对于放置在<BODY></BODY>之间的JavaScript，则需要把它放置在适当位置。当然，也可以把JavaScript放在表格中，这样做可以起到精确定位的作用。

2. 使用外部文件

另外一种插入 JavaScript 的方法，是把 JavaScript 代码写到另一个文件当中(此文件通常应该用“.js”作文件扩展名)，然后用格式为“<script src="javascript.js"></script>”的标签将它嵌入到文档中。注意，这时必须使用“</script>”标签。

<script>标签还有两个属性：language(缩写 lang)和 type，前者说明脚本所使用的语言，后者说明其类型。例如：<script language="JavaScript" type="text/JavaScript">。

3. 在浏览器中直接调用 JavaScript

在浏览器的地址栏中直接输入并执行 JavaScript 语句，方法如下：

```
javascript:<JavaScript 语句>
```

这种语句也可以直接放在 HTML 的超链接标签<a>中，代码如下：

```
<a href="javascript:<JavaScript 语句>">...</a>
```

6.1.5　一个简单的实例

【实例 6-1】一个简单的 JavaScript 实例

程序代码如 ex6_1.html 所示。

ex6_1.html

```
<HTML>
  <HEAD> <TITLE>一个简单的实例</TITLE>
  </HEAD>
  <BODY>
    <script language="javascript" >
      document.write("一个简单的实例");
    </script>
  </BODY>
</HTML>
```

如同 HTML 标记语言一样，JavaScript 程序代码也是由纯文本构成的，使用任何文本编辑软件都能编写。

JavaScript 代码由<script Language="javascript">...</script>说明。在标签<Script Language="JavaScript">...</Script>之间加入JavaScript脚本。本例中简单应用了JavaScript的document(文档)对象，执行其write()方法向浏览器显示区域输出内容。本例执行后浏览器中会显示“一个简单的实例”。

注意：

如果上面的例子不能运行，有可能是浏览器的安全设置不当造成的。在 IE 浏览器中，单击“工具”|“Internet”选项，在其中的“安全”选项卡中可以对脚本的运行条件进行设置。其他浏览器中类似，需要设置允许脚本运行。

6.2　JavaScript 基本语法

6.2.1　JavaScript 的语句

每一行 JavaScript 语句都有类似于以下的格式：

```
<语句>;
```

其中分号“;”是 JavaScript 语言作为一个语句结束的标识符。虽然当前大多数浏览器允许使用 Enter 键，但使用分号的习惯还是值得提倡的。

和语句不同，语句块(或被称为复合语句)是用大括号“{ }”括起来的一个或多个语句，语句块是允许嵌套的。

6.2.2　数据类型

JavaScript 中可用的数据类型如图 6-1 所示。

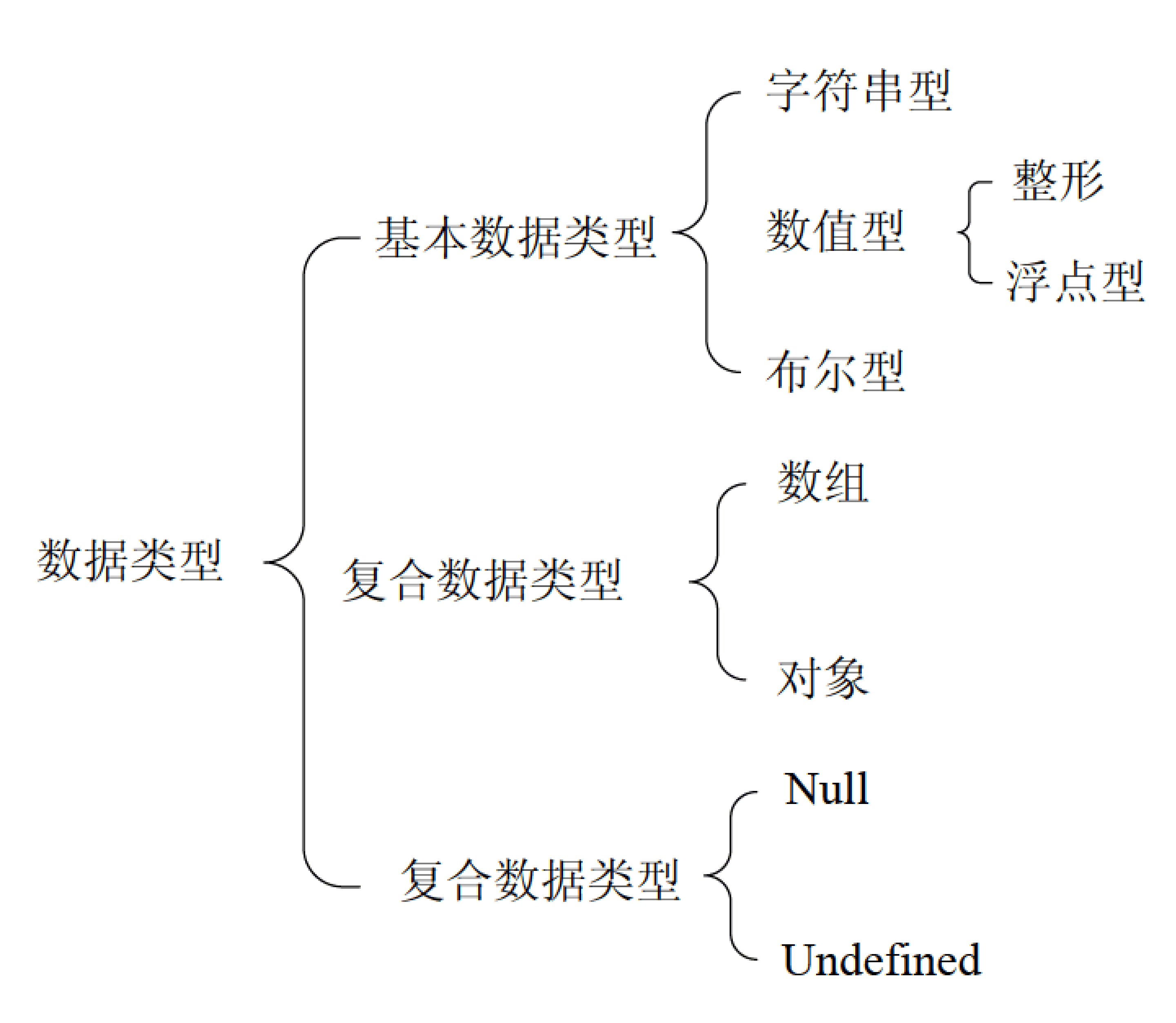


图 6-1　JavaScript 的数据类型

- 整形：只能储存整数。可以是正整数、0、负整数，可以是十进制、八进制、十六进制。八进制数的表示方法是在数字前加“0”，如“0123”表示八进制数“123”。十六进制则是加“0x”：“0xEF”表示十六进制数“EF”。
- 浮点型：即“实型”，能存储小数。一方面浮点数占用内存更多，且由于某些平台对浮点型数据的支持不稳定，因此若不必须可尽量不要使用浮点型。
- 字符串型：用引号“" "”或者“' '”表示的零个至多个字符，其中单引号或双引号均可，但是必须成对使用。单双引号可嵌套使用，如'这里是"JavaScript 教程"。不过 JavaScript 中引号的嵌套只能有一层。如果想嵌套多层，需要使用转义字符。

注意：

由于一些字符在屏幕上不能显示，或者 JavaScript 语法中已对这些字符定义了特殊用途，在使用这些字符时，就必须使用“转义字符”。转义字符用斜杠“\”开头：\'为单引号、\"为双引号、\n 为换行符、\r 为回车(以上只列出常用的转义字符)。使用转义字符，就可以做到引号多重嵌套，例如：'张三说："这是\"JavaScript 语言教程\"。"'

- 布尔型：常用于判断，只有两个值可选：true(表示“真”)和 false(表示“假”)。true 和 false 是 JavaScript 的保留字，属于常数。
- 对象：关于对象，在“对象化编程”部分中将详细介绍。
- 数组：是一种将相同数据类型的数据连续组织起来构成的一种数据结构。
- Null：Null 类型只有一个值：null。关键字 null 不能用作函数或变量的名称。包含 null 的变量包含“无值”或“无对象”。换句话说，该变量没有保存有效的数、字符串、boolean、数组或对象。可以通过给一个变量赋 null 值来清除变量的内容。和其他一些语言不同，null 与 0 不相等(在 C 和 C++中它们是相等的)。需要指出的是，JavaScript 中 typeof 运算符将报告 null 值为 Object 类型，而非类型 Null。这种潜在的混淆是为了向下兼容。
- Undefined：当对象属性不存在或声明了变量但从未赋值时，将得到 undefined 类型。

注意：

不能通过与 Undefined 做比较来测试一个变量是否存在，需要检测它的类型是否为“Undefined”。

6.2.3　变量

从字面上看，变量是可变的量；从运行的角度看，变量是用于存储数据的存储器。所存储的值，可以是数字、字符或其他类型的数据。

1. 变量的命名

对于变量的命名，要求其中只包含字母、数字和/或下划线；必须以字母开头；不能太长；不能与 JavaScript 保留字重复。其中保留字包括：表 6-1 所示的 JavaScript 的保留字以及表 6-2 所示的 JavaScript 为将来保留的关键词。

表 6-1　JavaScript 的保留字

break	delete	function	return	typeof
case	do	if	switch	var
catch	else	in	this	void
continue	false	instanceof	throw	while
debugger	finally	new	true	with
default	for	null	try	

表 6-2　JavaScript 为将来保留的关键词

abstract	double	goto	native	static
boolean	enum	implements	package	super
byte	export	import	private	synchronized
char	extends	int	protected	throws
class	final	interface	public	transient
const	float	long	short	volatile

JavaScript 是大小写敏感的，variable 和 Variable 是两个不同的变量，此外大部分命令和“对象”都是区分大小写的。为此，给变量命名时最好避免单个字母，如“a”“b”“c”等，最好使用能清楚表达该变量的单词，也便于自己及他人理解。

按照惯例，变量名及函数名一般小写，多个单词中除首单词外其余单词首字母大写，例如：myVariable 和 myAnotherVariable。

2. 变量的声明

由于采用动态编译，代码中的错误不易被发现；因此使用变量前声明变量可最大限度地发现代码中存在的错误，在声明时也可赋初值。声明变量的基本格式如下：

```
var <变量> [= <值>];
```

其中的 var 是用作变量的声明的保留字。最简单的声明方法就是“var <变量>;”，这将为<变量>准备内存，并给它赋初始值“null”。如果加上“= <值>”，则给<变量>赋予初始值<值>。以下是声明变量的一些例子：

```
var sum;                        // 单个声明
var red, green, blue;           // 用单个 var 声明的多个变量
var total = 0, iLoop = 100;     // 声明变量的同时进行初始化
```

当然，由于 JavaScript 采用了弱类型，如果未声明变量或者就算声明某个变量的类型，在后面的程序中仍然可以赋予其他类型的数据。

3. 变量的作用域

JavaScript 的变量有两种不同的作用域：全局的和局部的。

局部变量属于某个函数或语句块，每次进入该部分时都会创建和销毁这些变量，因而这些变量不能在区域外访问。局部变量可以和全局变量重名，但在区域内部时只有局部变量有效。

在任何函数外声明的都属于全局变量。如果在 var 语句中没有对变量初始化，则该变量会暂时成为 JavaScript 的 undefined 类型。对于 undefined 类型的变量，解释器将其作为全局变量。

4. 变量的赋值

可以在任何时候对变量赋值，赋值的方法是：

```
<变量>=<表达式>;
```

其中“=”叫“赋值符”，其作用是把右边的值赋给左边的变量。

6.2.4　运算符与表达式

1. 运算符

运算符可以是四则运算符、关系运算符、位运算符、逻辑运算符、复合运算符。表 6-3 将这些运算符从高优先级到低优先级进行排列，并对各种运算加以说明。

表 6-3　运算符及其优先级

运　算	实　例	运 算 说 明
括号	(x) [x]	中括号只用于指明数组的下标
求反、自加、自减	-x	返回 x 的相反数
	!x	返回与 x (布尔值)相反的布尔值
	x++	x 值加 1，但仍返回原来的 x 值
	x--	x 值减 1，但仍返回原来的 x 值
	++x	x 值加 1，返回后来的 x 值
	--x	x 值减 1，返回后来的 x 值
乘、除	x*y	返回 x 乘以 y 的值
	x/y	返回 x 除以 y 的值
	x%y	返回 x 与 y 的模(x 除以 y 的余数)
加、减	x+y	返回 x 加 y 的值
	x-y	返回 x 减 y 的值
关系运算	x<y x<=y x>=y x>y	当符合条件时返回 true 值，否则返回 false 值
等于、不等于	x==y	当 x 等于 y 时返回 true 值，否则返回 false 值
	x!=y	当 x 不等于 y 时返回 true 值，否则返回 false 值
位与	x&y	当两个数位同时为 1 时，返回的数据的当前数位为 1，其他情况都为 0
位异或	x^y	两个数位中有且只有一个为 0 时，返回 0，否则返回 1
位或	x|y	两个数位中只要有一个为 1，则返回 1；当两个数位都为零时才返回零
	位运算符通常会被当作逻辑运算符来使用。它的实际运算情况是：把两个操作数(即 x 和 y)转化成二进制数，对每个数位执行以上所列工作，然后返回得到的新二进制数。由于“真”值在计算机内部(通常)是全部数位都是 1 的二进制数，而“假”值则是全部是 0 的二进制数，所以位运算符也可以充当逻辑运算符	
逻辑与	x&&y	当 x 和 y 同时为 true 时返回 true，否则返回 false
逻辑或	x||y	当 x 和 y 任意一个为 true 时返回 true，当两者同时为 false 时返回 false

(续表)

运　　算	实　　例	运 算 说 明
		逻辑与/或有时候被称为“快速与/或”。这是因为当第一操作数(x)已经可以决定结果，它们将不去理会 y 的值。例如，false && y，因为 x == false，不管 y 的值是什么，结果始终是 false，于是本表达式立即返回 false，而不论 y 是多少，甚至 y 可以导致出错，程序也可以照样运行下去
条件	c?x:y	当条件 c 为 true 时返回 x 的值(执行 x 语句)，否则返回 y 的值(执行 y 语句)
赋值、复合运算	x=y	把 y 的值赋给 x，返回所赋的值
	x+=y　x-=y x*=y　x/=y x%=y	x 与 y 相加/减/乘/除/求余，所得结果赋给 x，并返回 x 赋值后的值

注意：

所有与四则运算有关的运算符都不能在字符串型变量上使用，可以使用+、+=来连接两个字符串。

2. 表达式

表达式与数学中的算式类似，是指具有一定值、用运算符将常数和变量连接起来的代数式，一个表达式可只包含一个常数或一个变量。

技巧：

一些用来赋值的表达式，由于有返回值，可以加以特殊利用，如 a=b=c=10，可以一次对三个变量赋值。

6.2.5　功能语句

JavaScript 基本编程命令，也称为“语句”。根据 JavaScript 语句的功能，可分为注释语句、条件语句和循环语句。

1. 注释语句

与大多数编程语言一样，JavaScript 的注释是被编译器忽略的。注释方式有两种：单行注释和多行注释。

- 单行注释用双反斜杠“//”表示。一旦出现了“//”，则其后面的部分将被忽略。
- 多行注释是用“/*”和“*/”括起来的一行到多行文字。程序执行到“/*”处，将忽略其后的所有文字，直到出现“*/”为止。

养成写注释的习惯，在自己或他人后期理解时能节省大量的时间。调试程序时，有时需要把一段代码换成另一段代码，或者暂时不要一段代码。这时也可以用注释，将它们“隐

藏”起来，等确认后再删除。

2. 条件语句

条件语句用于选择执行的功能，可以根据表达式的值，有条件地执行一组语句。其分为 if 语句和 switch 语句两种。

(1) if 语句

定义 if 语句的格式为：

```
if ( <条件> ) <语句 1> [ else <语句 2> ];
```

当<条件>为真时执行<语句 1>，否则，如果 else 存在的话，就执行<语句 2>。与条件表达式不同的是，if 语句不能返回数值。其中的<条件>是布尔值，必须用小括号括起来；<语句 1>和<语句 2>都只能是一个语句，如果希望使用多个语句，需用语句块。

对于 if 语句而言，必须注意 if 和 else 的匹配，代码示例如下：

```
if (a == 1)
   if (b == 0) alert(a+b);
else
   alert(a-b);
```

在 JavaScript 中，后期的版本对花括号的位置有严格的规定。if 语句的左括号必须与 if 在同一行。如果有 else 语句，则 if 语句的右括号必须与 else 语句块的左括号及 else 在同一行。

else 总是与其最近且未被匹配的 if 匹配，本代码企图用缩进的方法说明 else 是与 if(a==1)对应的，但是实际上，由于 else 与 if(b==0)最相近，本代码不是按作者的想法运行的。能正确表达作者意图的代码如下：

```
if (a == 1) {
   if (b == 0) alert(a+b);
} else {
   alert(a-b);
}
```

如果一行代码太长或者涉及复杂的嵌套，可考虑采用多行文本，如上例中 if(a==1)后面没有立即写上语句，而是换一行再继续写，浏览器是不会混淆的。此外，使用缩进也是很好的习惯，当一些语句与上面的语句具有从属关系时，缩进能提高程序的可读性，方便阅读、编写或后续的修改。

(2) switch 语句

如果要把某些数据分类，如将学生的成绩按优、良、中、差分类，若使用 if 语句，则程序可能如下：

```
if (score >= 0 && score < 60) {
   result = 'fail';
} else if (score < 80) {
   result = 'pass';
```

```
} else if (score < 90) {
   result = 'good';
} else if (score <= 100) {
   result = 'excellent';
} else {
   result = 'error';
}
```

这看起来没有问题，但由于 if 语句太多，程序看起来有点乱。switch 语句是解决此类问题最好的方法，使用 switch 语句的程序结构如下：

```
switch (e) {
   case r1: (// 注意：冒号)
      ...
      [break;]
   case r2:
      ...
      [break;]
   ...
    [default:
      ...]
}
```

其作用是：先计算 e 的值(e 为表达式)，然后跟下面“case”之后的 r1、r2……比较，当找到等于 e 的值时，则执行该“case”后的语句，直到遇到 break 语句或 switch 段落结束符“}”。若未找到匹配项，则执行“default:”后边的语句，若没有 default 块，则整个 switch 语句结束。下面的代码使用了 switch 语句来实现相同的功能：

```
switch (parseInt(score / 10)) {
   case 0:
   case 1:
   case 2:
   case 3:
   case 4:
   case 5:
      result = 'fail';
      break;
   case 6:
   case 7:
      result = 'pass';
      break;
   case 8:
      result = 'good';
      break;
   case 9:
      result = 'excellent';
      break;
   default:
      if (score == 100)
         result = 'excellent';
      else
```

```
        result = 'error';
}
```

其中 parseInt()方法的作用是取整。最后 default 段用的 if 语句，是为了处理 100 分的情况，因为 parseInt(100/10)为 10，而不是 9。

3. 循环语句

循环是程序中重要的结构。其特点是，若给定条件成立，则反复执行某程序段，直到条件不成立为止。给定的条件称为循环条件，反复执行的程序段称为循环体。JavaScript 提供了多种循环语句，使用它们可以构成不同形式的循环结构。循环语句包括：for 语句、while 语句和 do...while 语句。

(1) for 语句

for 语句的一般格式如下：

```
for (<变量>=<初始值>; <循环条件>; <变量累加方法>) <语句>;
```

该语句的作用是重复执行<语句>，直到<循环条件>为不成立为止。其运行顺序为：首先给<变量>赋<初始值>，其次*判断<循环条件>(应该是一个关于<变量>的条件表达式)是否成立，如果成立就执行<语句>，然后按<变量累加方法>对<变量>作累加，回到上面的“*”处重复，如果不成立就退出循环。下面是一个例子：

```
for (i = 1; i < 10; i++)
   document.write(i);
```

本语句先对变量 i 赋初始值 1，然后执行 document.write(i)语句(其作用是输出 i 的值)；循环时执行 i++(将 i 加 1)；循环直到 i<10 这个条件不满足为止，即结束时 i>=10。最终实现了在浏览器中输出“123456789”。

与 if 语句一样，<语句>的部分只能是一行语句，如果想用多条语句，需要用语句块。

与其他语言不同，JavaScript 的 for 循环没有规定循环变量每次循环一定要加一或减一，<变量累加方法>可以是任意的赋值表达式，如 i+=3、i*=2、i-=j 等都成立。

使用循环，能简化 HTML 文档中大量有规律重复的内容，从而提高网页下载的速度。

(2) while 语句

while 语句的一般格式如下：

```
while (<循环条件>) <语句>;
```

while 语句的作用是当满足<循环条件>时执行<语句>，可以看作 for 语句的简化。<语句>也同 for 语句中一样只能是一条语句，但是一般情况下为了改变循环变量，通常使用语句块。否则一旦形成“死循环”，就会给运行此代码的浏览器造成问题，例如，内存占用很大、因 CPU 高负荷而导致“假死机”现象等。

(3) do…while 语句

do…while 语句的一般格式如下：

```
do <语句> while (<循环条件>);
```

和 while 语句相似，与 while 语句循环的不同之处在于：它先执行循环中的语句，然后再判断表达式是否为真，如果为真则继续循环；如果为假，则终止循环。因此，do...while 循环至少要执行一次循环语句。

(4) break 和 continue

有时候在循环体内，需要立即跳出循环或跳过循环体内其余代码而进行下一次循环。这时 break 语句和 continue 语句可以完成。

将 break 语句置于循环体内，其作用是立即跳出本循环，用法如下：

```
break;
```

continue 语句位于循环体内，作用是中止本次循环，并执行下一次循环。如果循环的条件已经不符合，就跳出循环，其用法如下：

```
continue;
```

【实例 6-2】JavaScript 的循环实例

程序代码如 ex6_2.html 所示。

ex6_2.html

```
<script language="JavaScript" type="text/javascript">
<!--
  for (i = 1; i < 10; i++) {
    if (i == 2 || i == 4 || i == 7) continue;
  document.write(i);
}
//-->
</script>
```

运行后浏览器上显示出：1235689。

6.2.6　函数

“函数”是指能完成某种功能的代码块。常见的函数有：构造函数，如 Array()，能构造一个数组；全局函数，即全局对象里的方法；自定义函数，即用户自己构建的函数。

在 JavaScript 中，函数可以在脚本中被事件触发或被其他语句调用。一般在编写脚本时，当脚本较长时，可以考虑用一个或多个函数分解其中的重复部分，最终形成若干个功能更单一的函数。虽然这并非编写脚本的强制要求，但通过函数的运用，可以提高代码的重用性，使脚本更具可读性，也便于编写与调试。

与其他语言不同的是，JavaScript 并不区分函数(完成一定的功能并有返回值)和过程(完成一定的功能，但没有返回值)，在 JavaScript 中只有函数，函数完成一定功能后可以有返回值，也可以没有返回值，也就是说，JavaScript 函数涵盖了其他语言中的函数和过程。

JavaScript 中的构造函数用于建立并初始化特殊类型的对象，其名称和类的名称相同，用关键字 new 调用。通过构造函数可以创建空对象，然后构造函数负责为新对象执行相应的初始化(创建属性并赋予初始值)，最后构造函数返回所生成的对象。

JavaScript 包含很多内部函数。某些函数可以操作表达式和特殊字符，有些可将字符串转换为数值。比如：内部函数 eval()，可对以字符串形式表示的任意有效的表达式求值，这个函数有一个参数，该参数就是希望求得数值的字符串，下面给出一个使用本函数的示例具体程序代码如下：

```
var anExpression = "6 * 9 % 7";
var total = eval(anExpression); // 将变量 total 赋值为 5
var yetAnotherExpression = "6 * (9 % 7)";
total = eval(yetAnotherExpression) // 将变量 total 赋值为 12
// 将一个字符串赋给 totality (注意嵌套引用)
var totality = eval("'...surrounded by acres of clams.'");
```

JavaScript 允许用户自定义函数，定义函数使用如下方式：

```
function 函数名([参数集]) {
   ...
     [return[ <值>];]
}
```

值得注意的是：即使整个函数只有一个语句，在 function 之后及结尾的大括号也是不能省略的。函数名与变量名有一样的命名规则，即只能包含字母、数字、下划线，以字母开头、不能与保留字重复等。若没有参数，可以不写参数集，但外面的圆括号是必需的。

1. 参数

参数是向函数内部传递信息的桥梁。例如，如果希望求立方的函数返回 3 的立方，就需要有一个变量来接收“3”这个数值，这个特殊的变量就是参数。参数集是一个或多个用逗号分隔开来的参数的集合，如 a、b、c。

函数的语句中会包含“return”这个特殊的语句。执行一个函数的时候，一旦碰到 return 语句，函数立刻停止执行，并返回到调用它的程序中。如果“return”后带有<值>，则退出函数的同时返回该值。

```
function addAll(a, b, c) {
   return a + b + c;
}
var total = addAll(20, 40, 60);
```

上面的例子建立了 addAll()函数，它有 3 个参数：a、b、c，其作用是返回三个数相加的结果。在函数外部，利用“var total = addAll(20, 40, 60);”来接收该函数的返回值。

在函数的内部，参数可直接作为变量使用，并可用 var 语句来申明变量，但是这种变量不能被函数外部调用。要使函数内部的信息能被外部调用，要么使用“return”返回值，要么使用全局变量。

2. 全局变量

在脚本的“根部”(非函数内部)的“var”语句所定义的变量就是全局变量，它能在任意位置使用。

【实例 6-3】自定义函数实例

程序代码如 ex6_3.html 所示。

ex6_3.html

```
<HTML>
  <HEAD><TITLE>自定义函数实例</TITLE>
    <SCRIPT language="JavaScript">
      var epsilon = 0.00000000001; // 一些需要测试的极小数字
      // 测试整数的函数
      function integerCheck(a, b, c) {
          // 测试。
          if ( (a*a) == ((b*b) + (c*c)) )
            return true;
          else
            return false;
      }
      // 测试浮点数的函数。
      function floatCheck(a, b, c) {
          // 得到测试数值。
          var delta = ((a*a) - ((b*b) + (c*c)))
          // 测试需要绝对值
          delta = Math.abs(delta);
          // 如果差小于 epsilon，那么它相当接近
          if (delta < epsilon)
            return true;
          else
            return false;
      }
      // 三元检查。
      function checkTriplet(a, b, c) {
          // 创建临时变量，用于交换值
          var d = 0;
          // 先将最长的变量移动到位置“a”。需要的话交换 a 和 b
          if (b > a) {
              d = a; a = b; b = d;
          }
          // 需要的话交换 a 和 c
          if (c > a) {
              d = a; a = c; c = d;
          }
          // 测试全部的 3 个值，看其是否为整数？
          if (((a % 1) == 0) && ((b % 1) == 0) && ((c % 1) == 0)) {
              // 如果成立，使用精确检查。
              return integerCheck(a, b, c);
          }
```

```
            else {
               // 如果不成立，取尽可能相近的。
               return floatCheck(a, b, c);
            }
         }
         // 下面的三个语句赋给范例值，用于测试。
         var sideA = 3;
         var sideB = 4;
         var sideC = 5; //try float
         // 调用函数。调用后，'result' 中包含了结果
         var result = checkTriplet(sideA, sideB, sideC);
         if(result)
            document.write('成立！ ');
         else
            document.write('不成立！ ');
      </SCRIPT>
   </HEAD>
   <BODY>
   </BODY>
</HTML>
```

本例中的 checkTriplet()函数以三角形的边长为参数。通过查看三条边的长度是否可以组成一个毕达哥拉斯三元组(直角三角形斜边长度的平方等于其他两条边长的平方和)来计算该三角形是否为直角三角形。实际测试时 checkTriplet()函数要调用另外两个函数中的一个函数。

注意：

在浮点数测试版本中极小数(“epsilon”)作为测试变量使用。由于浮点运算的不确定性和舍入误差，除非问题中的三个值均已知为整数。

本实例运行后在屏幕上会显示“成立！”或“不成立”，读者可调整 sideA、sideB、sideC 的值，来测试这个程序并观察运行的结果。

6.3　对象化编程

JavaScript 是支持“对象化编程”的，或者叫“面向对象编程”的。所谓“对象化编程”，意思是将编程所涉及的组成部分划分成对象，对象还可以划分为更小的对象直至不能进一步划分为止。JavaScript 编程方法就以对象为出发点，小到一个变量，大到整个网页的文档、窗口甚至屏幕，都是对象，本节从“基于对象”的角度介绍 JavaScript。

6.3.1　对象的基本知识

虽然 JavaScript 只是基于对象的，但它能创建用户自定义对象。通过这个方式，可以扩大 JavaScript 的应用范围，编写功能强大的 Web 应用。

对象是属性和方法的集合。对象是可以从 JavaScript 代码中划分出来的一部分，可以是一段文字、一幅图片、一个表单(Form)等。每个对象具有属性(properties)、方法(methods)和事件(event)。对象的属性反映该对象某些特定的性质，例如，字符串的长度、图像的长宽、文字框(Textbox)里的文字等；对象的方法能对该对象实施特定操作，例如，表单的“提交”(Submit)，窗口的“滚动”(Scrolling)等；而对象的事件能响应发生在对象上的操作，例如，单击提交表单产生表单的“提交事件”，单击链接产生“单击事件”。不是所有的对象都必须有以上三个性质，如有些没有事件，有些只有属性。

1. JavaScript 对象的基本组成

JavaScript 中的对象是由属性和方法两种基本元素构成的。前者是对象在实施其所需要行为的过程中，实现信息的装载单位，从而与变量相关联，可以理解为数据；后者是指对象能够按照设计者的意图而被触发执行，与特定函数相关联的机制，可以理解为算法。当然，对象化编程不是数据＋算法，而是通过封装机制，将它们有机地融合在一起，并提供了更多的管理机制。

2. 对象的创建

若希望使用对象，首先需要获得对象，可采用以下几种方式获得对象：引用 JavaScript 的内部对象、由浏览器环境所提供的对象或自行创建对象。

3. 有关对象操作语句

JavaScript 不是完全面向对象的语言，但在 JavaScript 中还是提供了若干操作对象的语句、关键词及运算符，现简要介绍如下。

(1) for...in 语句

使用方式为：for(对象属性名 in 已知对象名)

其功能是对已知对象的所有属性进行遍历，这种方式可不使用计数器，因此其优点是无须知道对象中属性的个数即可进行操作。

例如，下列程序代码显示了数组的内容。

```
for (var i=0; i<30;i++)
   document.write(object[i]);
```

这种方法通过数组的下标来访问，因此必须预先知道下标的范围。而使用 for...in 语句则更为方便，程序代码如下：

```
for(var prop in object)
   document.write(object[prop]);
```

运行时，在循环体中可自动将所有值取出来，直到最后一个为止。

(2) with 语句

使用该语句的意思是：在该语句体内，任何对变量的引用被认为是这个对象的属性，可减少代码量，程序代码如下：

```
with object{
...}
```

所有在 with 语句后的花括号中的语句，都已具有 object 对象的作用域。

(3) this 关键词

this 是对当前对象的引用。由于对象的引用是多层次、多方位的，多次的引用可能会造成混乱，即不清楚现在引用的是哪一个对象，为此 JavaScript 提供了一个指定当前对象的关键词 this。

(4) new 运算符

使用 new 运算符可以创建新对象，其程序代码如下：

```
newObject=new Object(parameters table);
```

其中 newObject 是所创建的新对象：object 是已经存在的类型；parameters table 是参数表。如希望创建一个日期类型的新对象，则可以使用如下代码：

```
newData=New Data()
birthday=New Data (December 12.1998)
```

后继的语句就可以使用 newData 和 birthday 这两个新对象了。

(5) 对象属性的引用

对象属性的引用可由下列 3 种方式之一实现。

1) 使用点(.)运算符。和大多数面向对象语言相同，引用对象的属性可用“<对象名>.<性质名>”的方法。如下所示：

```
city.ProvinceName=“江苏”
city.Name=“南京”
city.Date="1912"
```

其中 city 是一个已经存在的对象，ProvinceName、Name、Date 是它的三个属性，上面的操作完成了对其的赋值。

2) 通过对象的下标实现引用，程序代码如下：

```
city[0]=“南京”
city[1]=“昆明市”
city[2]=" 1912"
```

3) 以字符串的形式引用，程序代码如下：

```
city ["ProvinceName "]=“江苏”
city ["Name "]=“南京”
city ["Date"]="1912"
```

如上文所述，既然可以通过数组来访问属性，就可以采用 for...in 语句，以便在不知其具体个数的情况下访问，程序代码如下：

```
for (var prop in this)
```

```
docament.write(this[prop]);
```

(6) 对象方法的调用

在 JavaScript 中调用对象方法的方式为：

```
ObjectName.methods()
```

如引用 city 对象中的 show ()方法并将结果打印出来，则可使用代码如下：

```
document.write (university.show ())
```

或：

```
document.write(university)
```

如需要引用内部对象 math 中的 cos()函数，则程序代码如下：

```
with(math)
document.write(cos(30));
document.write(cos(60));
```

若不使用 with 则引用时会稍微复杂些，代码如下：

```
document.write(math.cos(30))
document.write(math.sin(60))
```

6.3.2　事件处理

事件处理是对象化编程中一个重要的环节，缺少了事件处理，程序会变得呆板，缺乏灵活性。

事件处理的过程是：发生事件→启动事件处理程序→事件处理程序作出反应。其中，要使事件处理程序得以启动，必须事先告诉对象，如果发生了什么事情，要进行什么样的操作，否则这个流程就不能进行下去。事件的处理程序可以是任意的 JavaScript 语句，但一般用特定的自定义函数。JavaScript 的事件处理模型如图 6-2 所示。

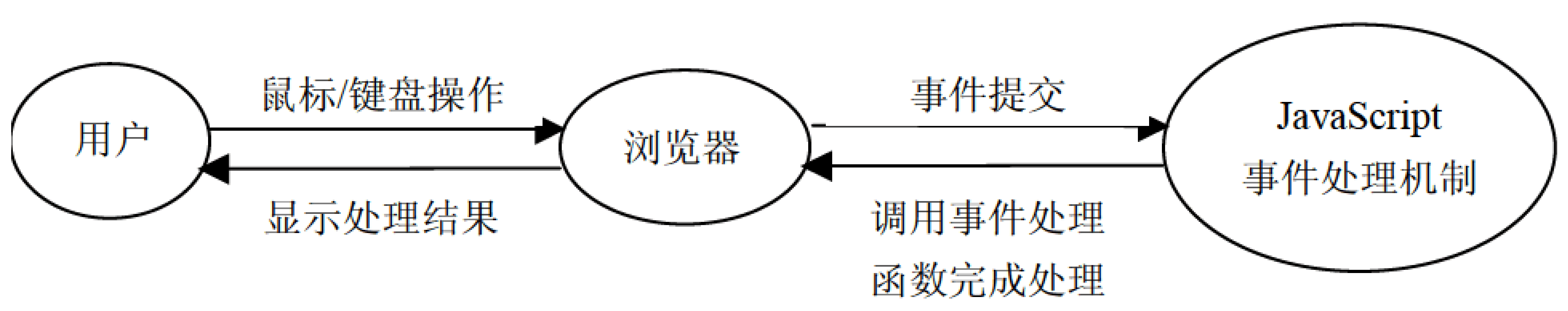


图 6-2　JavaScript 的事件处理模型

1. 指定事件处理程序

指定事件处理程序通常有以下 3 种方法。

(1) 直接在 HTML 标签中指定

这种方法是最普遍的，用法如下：

```
<标签 ...事件="事件处理程序" [事件="事件处理程序" ...]>
```

例如：

```
<BODY onload="alert('网页读取已经完成，请欣赏！')" onunload="alert('即将退出，再见！')">
```

上面的<BODY>标签，能在文档装载完毕时弹出一个对话框，显示“网页读取已经完成，请欣赏！”；在用户退出文档(或者关闭窗口，或者到另一个页面去)的时候弹出“即将退出，再见！”。当然，利用这个机制也可以实现在关闭窗口时弹出一个新窗口的操作。

(2) 为特定对象/事件编写一段 JavaScript 程序

这种方法用得较少，但是为了代码清晰或某些其他需要，有时也被采用。使用方法如下：

```
<script language="JavaScript" for="对象" event="事件">
...
(事件处理程序代码)
...
</script>
```

又如：

```
<script language="JavaScript" for="window" event="onload">
  alert('网页读取已经完成，请欣赏！');
</script>
```

(3) 在 JavaScript 语句中指定

其方法如下：

```
  <对象>.<事件> = <事件处理程序>;
```

用这种方法要注意的是，“事件处理程序”是真正的代码，而不是字符串形式的代码。如果事件处理程序是一个自定义函数，如没有参数，就可以不加“()”，例如：

```
function encounterError() {
  return true;
}
...
window.onerror = encounterError; // 没有使用“()”
```

这个例子将 encounterError()函数定义为出现 window 对象 onerror 事件后的处理程序。其效果是忽略该 window 对象下任何错误。但是在这种错误处理模式中，由引用不允许访问的 location 对象所产生的“没有权限”错误是不能忽略的。

2. 常用事件

各类常用事件介绍如下。

- onblur 事件：发生在窗口失去焦点的时候；主要应用于 window 对象。
- onchange 事件：发生在文本输入区的内容被更改，然后焦点从文本输入区移走之后，捕捉此事件主要用于实时检测输入内容的有效性，或者立刻改变文档内容；

可用于 Password 对象、Select 对象、Text 对象和 Textarea 对象等。

- onclick 事件：发生在对象被单击的时候。单击是指鼠标停留在对象上，按下鼠标键，没有移动鼠标而放开鼠标键这一个完整的过程；一个普通按钮对象(Button)通常具有 onclick 事件处理程序，因为这种对象根本不能从用户那里得到任何信息，没有 onclick 事件处理程序就没有任何用处。在按钮上添加 onclick 事件处理程序，可以模拟“另一个提交按钮”，方法是：在事件处理程序中更改表单的 action、target、encoding、method 等一个或几个属性，然后调用表单的 submit()方法。可应用于 Button 对象、Checkbox 对象、Image 对象、Link 对象、Radio 对象、Reset 对象和 Submit 对象等。

注意：

在 Link 对象的 onclick 事件处理程序中返回 false 值(return false)，能阻止浏览器打开此链接。即如果有一个这样的链接：Go!，那么无论用户怎样点击，都不会转向 www.a.com 网站，除非用户禁止浏览器运行 JavaScript。

- onerror 事件：发生在错误发生的时候。它的事件处理程序通常叫作“错误处理程序”(Error Handler)，用来处理错误。可应用于 window 对象。
- onfocus 事件：发生在窗口得到焦点的时候，常应用于 window 对象。
- onload 事件：发生在文档全部装载完毕的时候。这意味着不仅是 HTML 文件，而且包含图片、插件、控件、小程序等全部内容的装载。本事件是 window 事件，但在 HTML 中指定事件处理程序的时候，将它写在<BODY>标签中，可应用于 window 对象。
- onmousedown 事件：发生在用户把鼠标放在对象上按下鼠标键的时候，可以参考 onmouseup 事件，应用于 Button 对象和 Link 对象。
- onmouseout 事件：发生在鼠标离开对象的时候，可参考 onmouseover 事件；可应用于 Link 对象。
- onmouseover 事件：发生在鼠标进入对象范围的时候。这个事件和 onmouseout 事件，再加上图片的预读，就可以做到当鼠标移到图像链接上，图像更改的效果了。有时会看到当鼠标指向某个链接时，状态栏上不显示地址，而显示其他的资料，看起来这些资料是可以随时更改的，此功能的实现代码如下：

```
<a href="..."   onmouseover="window.status='Click Me Please!'; return true;"
  onmouseout="window.status="; return true;">
```

它可应用于 Link 对象。

注意：

鼠标激活链接(onmouseover)是 Web 应用最广泛和最有效的动态方法之一，其原因在于，它能使用户获得清晰、直接的反馈。设想将鼠标移到一个超文本链接上，该链接将会变为高亮度显示、改变颜色或者产生其他的变化以表示“这是一个链接!”。但这并不是说

所创建的所有鼠标激活链接(mouseover)都是相同的。最糟糕的情况是用 Java 等语言来实现，如 Macromedia 的 Shockwave 格式，它们的执行需要在浏览器中安装特定插件。因此最好利用各种浏览器都支持的 JavaScript 来编写。

- onmouseup 事件：发生在用户把鼠标放在对象上鼠标键被按下之后放开鼠标键时。如果按下鼠标键的时候，鼠标并不在放开鼠标的对象上，则本事件不会发生；可应用于 Button 对象和 Link 对象。
- onreset 事件：发生在表单的“重置”按钮被单击(按下并放开)的时候，通过在事件处理程序中返回 false 值(return false)可以阻止表单重置；常用于 Form 对象。
- onresize 事件：发生在窗口被调整大小的时候，可应用于 window 对象。
- onsubmit 事件：发生在表单的“提交”按钮被单击(按下并放开)的时候，可以使用该事件来验证表单的有效性。通过在事件处理程序中返回 false 值(return false)可以阻止表单提交，可应用于 Form 对象。
- onunload 事件：发生在用户退出文档(或者关闭窗口，或者到另一个页面去)的时候；与 onload 一样，需要写在<BODY>标签中。有的网站用这个方法来弹出“调查表单”，以“强迫”用户填写；而有的则利用它弹出广告窗口。因此这种“onunload="open..."”的方法很不好，有时甚至会因为弹出太多窗口而导致系统资源缺乏；该事件可应用于 window 对象。

6.3.3　JavaScript 的内部对象

JavaScript 本身提供了一些有用的内部对象和方法，用户可以直接使用。

JavaScript 语言中提供了诸如 String(字符串)、Math(数值计算)和 Date(日期)等多种对象和其他一些相关的属性和方法，为编程人员快速开发功能强大的应用提供了可能。在 JavaScript 对象属性与方法的引用中，有两种情况：一种是该对象为静态对象，即在引用该对象的属性或方法时不需要为它创建实例；而另一种对象则是在引用它的对象或方法时必须为它创建一个实例，即该对象是动态对象。

对于 JavaScript 内部对象的引用，是紧紧围绕着它的属性与方法进行的，因而明确对象的静态、动态特性对于掌握和理解 JavaScript 对象具有重要意义。以下简要介绍 JavaScript 常用对象。

1. Number“数字”对象

这个对象用得较少，不过属于“Number”的成员有很多。

(1) 常用属性

几种常用属性的含义如下。

- MAX_VALUE：Number.MAX_VALUE；返回“最大值”。
- MIN_VALUE：Number.MIN_VALUE；返回“最小值”。
- NaN：Number.NaN 或 NaN；返回“NaN”。
- NEGATIVE_INFINITY：Number.NEGATIVE_INFINITY；返回负无穷大，即比“最

小值”还小的值。

- POSITIVE_INFINITY：Number.POSITIVE_INFINITY；返回正无穷大，即比“最大值”还大的值。

(2) 常用方法。Number“数字”对象常用方法为 toString():，具体事宜方法如下。

toString()：<数值变量>.toString()；返回数值的字符串形式。例如：若 a=123；则 a.toString()='123'。

2. String 字符串对象

String 对象属于内部对象，且为静态的，在声明一个字符串对象时最简单、快捷、有效且常用的方法就是直接赋值。

(1) 属性

该对象只有一个属性，即 length。它表明了字符串中的字符个数，包括所有符号。使用方法如下：

```
<字符串对象>.length
```

例如：

```
mytest="This is a JavaScript"
mystringlength=mytest.length
```

最后 mystringlength 返回 mytest 字符串的长度为 20。

(2) 常用方法

string 对象的方法共有 19 个。主要用于有关字符串在 Web 页面中的显示、字体大小、字体颜色、字符的搜索以及字符的大小写转换。其中常用的方法如下。

- 锚点 anchor()：该方法创建 HTML 文件中的 anchor 标签。可用下列方式访问：string.anchor(anchorName)。
- 有关字符显示的控制方法：Italics()斜体字显示，bold()粗体字显示，blink()字符闪烁显示，small()字符用小体字显示，fixed()固定高亮字显示，fontsize(size)控制字体大小等。
- 字体颜色：fontcolor(color)。
- 字符串大小写转换：toLowerCase()小写转换，toUpperCase()大写转换。例如，将一个给定的串分别转换成大写和小写格式，代码如下：

```
string=stringValue.toUpperCase
string=stringValue.toLowerCase
```

- 字符搜索：indexOf([charactor,fromIndex])，从指定 formIndtx 位置开始搜索 charactor 第一次出现的位置。
- 返回字符串的一部分字符串：substring(start,end)，将从 start 开始到 end 的字符全部返回。

- 读取字符：charAt()，用法：<字符串对象>.charAt(<位置>)；可返回该字符串位于第<位置>位的单个字符。注意：字符串中的一个字符是第 0 位的，第二个才是第 1 位的，最后一个字符是第 length−1 位的。
- 读取字符的编码：charCodeAt()，用法：<字符串对象>.charCodeAt(<位置>)；返回该字符串位于第<位置>位的单个字符的 ASCII 码。
- 字符编码：fromCharCode()，用法：String.fromCharCode(a, b, c...)；返回一个字符串，该字符串每个字符的 ASCII 码由 a,b,c...等来确定。
- 字符串查找：indexOf()，用法：<字符串对象>.indexOf(<另一个字符串对象>[, <起始位置>])；该方法从<字符串对象>中查找<另一个字符串对象>(如果给出<起始位置>就忽略之前的位置)，如果找到了，就返回它的位置，没有找到就返回“−1”。所有的“位置”都是从零开始的。
- 字符串反向查找：lastIndexOf()，用法：<字符串对象>.lastIndexOf(<另一个字符串对象>[, <起始位置>])；跟 indexOf()相似，不过是从后边开始查找。
- 分隔字符串：split()，用法：<字符串对象>.split(<分隔符字符>)；返回一个数组，该数组是从<字符串对象>中分离开来的，<分隔符字符>决定了分离的地方，它本身不会包含在所返回的数组中。例如，'1&2&345&678'.split('&')返回数组：1,2,345,678。
- 按位置取子字符串：substring()，用法：<字符串对象>.substring(<始>[,<终>])；返回原字符串的子字符串，该字符串是原字符串从<始>位置到<终>位置的前一位置的一段。<终> − <始> = 返回字符串的长度(length)。如果没有指定<终>或指定得超过了字符串长度，则子字符串从<始>位置一直取到原字符串尾。如果所指定的位置不能返回字符串，则返回空字符串。
- 按长度取子字符串：substr()，用法：<字符串对象>.substr(<始>[,<长>])；返回原字符串的子字符串，该字符串是原字符串从<始>位置开始，长度为<长>的一段。如果没有指定<长>或指定得超过了字符串长度，则子字符串从<始>位置一直取到原字符串尾。如果所指定的位置不能返回字符串，则返回空字符串。
- 转换为小写：toLowerCase()，用法：<字符串对象>.toLowerCase()；返回把原字符串所有大写字母都变成小写的字符串。
- 转换为大写：toUpperCase()，用法：<字符串对象>.toUpperCase()；返回把原字符串所有小写字母都变成大写的字符串。

3. Array 数组对象

数组对象是一个对象的集合，且可以是不同类型的。数组的每一个成员对象都有一个“下标”，用来表示它在数组中的位置，下标从 0 开始递增。

数组的定义方法：var <数组名> = new Array();。这个语句定义了一个空数组。

之后若要添加数组元素，就用<数组名>[<下标>] = ...;。

注意：

这里的方括号不是“可以省略”的意思，数组的下标表示方法就是用方括号括起来的。如果想在定义数组的时候直接初始化数据，请用：

```
var <数组名> = new Array(<元素 1>, <元素 2>, <元素 3>...);
```

例如：var myArray = new Array(1, 4.5, 'Hi')；定义了一个数组 myArray，其中的元素是：myArray[0] == 1; myArray[1] == 4.5; myArray[2] == 'Hi'。

但是，如果元素列表中只有一个元素，而这个元素又是一个正整数的话，这将定义一个包含<正整数>个空元素的数组。

注意：

JavaScript 只有一维数组。千万不要尝试使用“Array(3,4)”来定义一个 3 × 4 的二维数组，或者用“myArray[2,3]”这种方法来返回“二维数组”中的元素。任意“myArray[...,3]”这种形式的调用其实只返回了“myArray[3]”。

要使用多维数组，只有使用这种变通的方法：

```
var myArray = new Array(new Array(), new Array(), new Array(), ...);
```

其实这是一个一维数组，其中的每一个元素又是一个数组。调用这个“二维数组”的元素的形式为：myArray[2][3] = ...;。

(1) 常用属性

- length：<数组对象>.length;
- 返回值：数组的长度，即数组里有多少个元素。它等于数组里最后一个元素的下标加一。所以，想添加一个元素，只需要 myArray[myArray.length] = ...即可。

(2) 常用方法

- 连接：join()，用法：<数组对象>.join(<分隔符>)；返回一个字符串，该字符串把数组中的各个元素串起来，用<分隔符>置于元素与元素之间。这个方法不影响数组原本的内容。
- 倒序：reverse()，用法：<数组对象>.reverse()；使数组中的元素顺序反过来。如果对数组[1, 2, 3]使用这个方法，它将使数组变成[3, 2, 1]。
- 子串：slice()，用法：<数组对象>.slice(<始>[, <终>])；返回一个数组，该数组是原数组的子集，始于<始>，终于<终>。如果不给出<终>，则子集一直取到原数组的结尾。
- 排序：sort()，用法：<数组对象>.sort([<方法函数>])；使数组中的元素按照一定的顺序排列。如果不指定<方法函数>，则按字母顺序排列。在这种情况下，80 是排在 9 之前的。如果指定<方法函数>，则按<方法函数>所指定的排序方法排序。这个方法函数有两个参数，分别代表每次排序比较时的两个数组项。sort()排序时对数据每次比较时都执行这个函数，当函数返回值为 1 的时候就交换两个数组项的

顺序，否则就不交换。

【实例 6-4】数组的排序

程序代码如 ex6_4.html 所示。

ex6_4.html

```
<script language="JavaScript" type="text/javascript">
<!--
    var arrA = [6,2,4,3,5,1];
    function desc(x,y) {
        if (x > y)
            return -1;
        if (x < y)
            return 1;
    }
    function asc(x,y) {
        if (x > y)
            return 1;
        if (x < y)
            return -1;
    }
    arrA.sort(desc);     // 按降序排序
    document.writeln(arrA);
    document.writeln("<br>");
    arrA.sort(asc);      // 按升序排序
    document.writeln(arrA);
//-->
</script>
```

该程序运行后的结果为：

```
6,5,4,3,2,1
1,2,3,4,5,6
```

4. Math 算术对象

Math 对象提供对数据的数学计算，提供了除加、减、乘、除以外的一些运算，如对数、平方根等，属于静态对象。

(1) 属性

Math 中提供了 6 个属性，它们是数学中经常用到的常数 E 、以 10 为底的自然对数 LN10、以 2 为底的自然对数 LN2、3.14159 的 PI、1/2 的平方根 SQRT1-2 和 2 的平方根为 SQRT2 等。Math 对象常用的属性及其含义如表 6-4 所示。

表 6-4　Math 的属性及其含义

属 性 名 称	含　义
E	常数 e (2.718281828...)
LN2	2 的自然对数 (ln 2)

(续表)

属 性 名 称	含　　义
LN10	10 的自然对数 (ln 10)
LOG2E	以 2 为底的 e 的对数 (log2e)
LOG10E	以 10 为底的 e 的对数 (log10e)
PI	π (3.1415926535...)
SQRT1_2	1/2 的平方根
SQRT2	2 的平方根

(2) 方法

Math 对象常用的方法及其含义如表 6-5 所示。

表 6-5　Math 的方法及其含义

属 性 名 称	含　　义
abs(x)	返回 x 的绝对值
acos(x)	返回 x 的反余弦值(余弦值等于 x 的角度)，用弧度表示
asin(x)	返回 x 的反正弦值
atan(x)	返回 x 的反正切值
atan2(x, y)	返回复平面内点(x, y)对应的复数的辐角，用弧度表示，其值在 -π 到 π 之间
ceil(x)	返回大于等于 x 的最小整数
cos(x)	返回 x 的余弦
exp(x)	返回 e 的 x 次幂 (ex)
floor(x)	返回小于等于 x 的最大整数
log(x)	返回 x 的自然对数 (ln x)
max(a, b)	返回 a, b 中较大的数
min(a, b)	返回 a, b 中较小的数
pow(n, m)	返回 n 的 m 次幂 (nm)
random()	返回大于 0 小于 1 的一个随机数
round(x)	返回 x 四舍五入后的值
sin(x)	返回 x 的正弦
sqrt(x)	返回 x 的平方根
tan(x)	返回 x 的正切

5. Date 日期对象

Date 对象提供了一个有关日期和时间的对象，它不属于静态对象，即必须使用 new 运算符创建一个实例后才能使用，例如：

```
MyDate=New Date();
```

Date 对象没有提供直接访问的属性，只有获取和设置日期和时间的方法。Date 对象能表示的日期以 1770 年 1 月 1 日 00:00:00 为基准，所表示的日期范围约等于该基准日期前后各 285,616 年；这个对象可以储存任意一个日期，并且可以精确到毫秒数(1/1000 秒)；所有日期时间，如果不指定时区，都采用“UTC”(世界时)时区，它与“GMT”(格林威治时间)在数值上是一样的。

定义一个日期对象的方法如下：

```
var d = new Date;
```

这个方法使 d 成为日期对象，并且已有初始值：当前时间。如果要自定义初始值，可以用如下代码：

```
var d = new Date(99, 10, 1);          //99 年 10 月 1 日
var d = new Date('Oct 1, 1999'); //99 年 10 月 1 日
```

此外还有一些方法，但最好的方法就是用下面介绍的“方法”来严格定义时间。以下有很多“g/set[UTC]XXX”这样的方法，它表示既有“getXXX”方法，又有“setXXX”方法。“get”是获得某个数值，而“set”是设定某个数值。如果带有“UTC”字母，则表示获得/设定的数值是基于 UTC 时间的，没有则表示基于本地时间或浏览器默认的时间。

- g/set[UTC]FullYear()：返回/设置年份，用四位数表示。如果使用“x.set[UTC]FullYear(99)”，则年份被设定为 0099 年。
- g/set[UTC]Year()：返回/设置年份，用两位数表示。设定的时候浏览器自动加上“19”开头，故使用“x.set[UTC]Year(00)”把年份设定为 1900 年。
- g/set[UTC]Month()：返回/设置月份。
- g/set[UTC]Date()：返回/设置日期。
- g/set[UTC]Day()：返回/设置星期，0 表示星期天。
- g/set[UTC]Hours()：返回/设置小时数，24 小时制。
- g/set[UTC]Minutes()：返回/设置分钟数。
- g/set[UTC]Seconds()：返回/设置秒钟数。
- g/set[UTC]Milliseconds()：返回/设置毫秒数。
- g/setTime()：返回/设置时间。该时间就是日期对象的内部处理方法：从 1970 年 1 月 1 日零时整开始计算到日期对象所指的日期的毫秒数。如果要使某日期对象所指的时间推迟 1 小时，就用“x.setTime(x.getTime() + 60 * 60 * 1000);”(1 小时 60 分，1 分 60 秒，1 秒 1000 毫秒)。
- getTimezoneOffset()：返回日期对象采用的时区与格林威治时间所差的分钟数。在格林威治东方的市区，该值为负，例如：中国时区(GMT+0800)返回“−480”。
- toString()：返回一个字符串，描述日期对象所指的日期。这个字符串的格式类似于“Fri Jul 21 15:43:46 UTC+0800 2000”。
- toLocaleString()：返回一个字符串，描述日期对象所指的日期，用本地时间表示格式，如“2000-07-21 15:43:46”。

- toGMTString()：返回一个字符串，描述日期对象所指的日期，用 GMT 格式。
- toUTCString()：返回一个字符串，描述日期对象所指的日期，用 UTC 格式。
- parse()：Date.parse(<日期对象>)；返回该日期对象的内部表达方式。

6. JavaScript 中的系统函数

JavaScript 中的系统函数又称内部方法。它提供了与任何对象无关的系统函数，使用这些函数不需创建任何实例，可直接使用。

- 返回字符串表达式中的值：eval(字符串表达式)，例如：test=eval("8+9+5/2");。
- 返回字符串 ASCI 码：unEscape (string)。
- 返回字符的编码：escape(character)。
- 返回实数：parseFloat(floatstring);。
- 返回不同进制的数：parseInt(numberstring ,radix)。

其中 radix 是数的进制，numberstring 字符串形式的数值。

7. JavaScript 的全局对象

全局对象是虚拟出来的，其目的在于将全局函数"对象化"。在 Microsoft JavaScript 语言参考中，它叫作"Global 对象"，但是引用它的方法和属性从来不用"Global.xxx"(况且这样做会出错)，而直接用"xxx"。

除了属性 NaN 以外，JavaScript 的全局对象还包含了下面的一些方法。

- eval()：把括号内的字符串当作标准语句或表达式来运行。
- isFinite()：如果括号内的数字是"有限"的(在 Number.MIN_VALUE 和 Number.MAX_VALUE 之间)就返回 true，否则返回 false。
- isNaN()：如果括号内的值是"NaN"则返回 true，否则返回 false。
- parseInt()：返回把括号内的内容转换成整数之后的值。如果括号内是字符串，则字符串开头的数字部分被转换成整数；如果以字母开头，则返回"NaN"。
- parseFloat()：返回把括号内的字符串转换成浮点数之后的值，字符串开头的数字部分被转换成浮点数；如果以字母开头，则返回"NaN"。
- toString()：<对象>.toString()；把对象转换成字符串。如果在括号中指定一个数值，则转换过程中所有数值将转换成特定进制。
- escape()：返回括号中的字符串经过编码后的新字符串。该编码应用于 URL，将空格写成"%20"格式。"+"不被编码，如果要"+"也被编码，请用：escape('...', 1)。
- unescape()：是 escape()的反过程。将括号中字符串解码为一般字符串。

6.3.4　JavaScript 的自定义类及对象

JavaScript 的内部对象提供了常用的一些功能，但如果需要建立特殊的对象，可以使用 JavaScript 的自定义类来完成。

1. 理解类的实现机制

在 JavaScript 中可以使用 function 关键字来定义一个“类”，然后再为类添加属性和方法。通常，在函数内通过 this 指针引用的变量或者方法都会成为类的成员。

【实例 6-5】自定义类实例

程序代码如 ex6_5.html 所示。

ex6_5.html

```
<script language="JavaScript" type="text/javascript">
<!--
   function class1(){
      var s="abc";
      this.p1=s;
      this.method1=function(){
         alert("this is a test method");
      }
   }
   var obj1=new class1();
   obj1.method1();
//-->
</script>
```

本实例运行后在浏览器中弹出一个对话框，其中显示“this is a test method”。

代码通过 new class1()获得对象 obj1，它自动获得了属性 p1 和方法 method1。JavaScript 规定对 function 本身的定义就是类的构造函数。

2. 通过 new 创建对象

结合上文所介绍关于对象的性质及 new 操作符的用法，用 new 创建对象的步骤如下：

(1) 当解释器遇到 new 操作符时便创建一个空对象。

(2) 开始运行 class1 这个函数，并将其中的 this 指针都指向这个新建的对象。

(3) 因为当给对象不存在的属性赋值时，解释器就会为对象创建该属性。例如，在 class1 中，当执行到 this.p1=s 这条语句时，就会添加一个属性 p1，并把变量 s 的值赋给它，这样函数执行就是初始化这个对象的过程，即实现构造函数的作用。

(4) 当函数执行完后，new 操作符就返回初始化后的对象。

这就是 JavaScript 所实现的面向对象的基本机制。由此可见，function 的定义实际上就是实现一个对象的构造器，这是通过函数来完成的。但是这种实现方式存在缺点，具体如下：

(1) 将所有的初始化语句、成员定义都放到一起，代码逻辑不够清晰，不易实现复杂的功能，如果实现了比较复杂的功能，那么代码通常显得凌乱。

(2) 每创建一个类的实例，都要执行一次构造函数，构造函数中定义的属性和方法总被重复地创建。

例如上面例子中所使用的代码截取如下：

```
this.method1=function(){
    alert("this is a test method");
}
```

此处的 method1 每创建一个 class1 实例，都会被创建一次，造成内存的浪费。而另一种类定义的机制 prototype 对象，就可以解决上述缺点。

3. 用 prototype 对象定义类成员

当新建一个 function 时，该对象的成员将自动赋给所创建的对象，读者可参考下面的实例来理解。

【实例 6-6】用 prototype 对象定义类成员

程序代码如 ex6_6.html 所示。

ex6_6.html

```
<script language="JavaScript" type="text/javascript">
<!--
    // 定义一个只有一个属性 prop 的类
    function class1(){
        this.prop=1;
    }
    // 使用函数的 prototype 属性给类定义新成员
    class1.prototype.showProp=function(){
        alert(this.prop);
    }
    // 创建 class1 的一个实例
    var obj1=new class1();
    // 调用通过 prototype 原型对象定义的 showProp 方法
    obj1.showProp();
//-->
</script>
```

本实例运行后在浏览器中弹出一个对话框，其中显示“1”。

prototype 是一个 JavaScript 对象，可以为 prototype 对象添加、修改、删除方法和属性，从而为一个类添加成员定义。

了解了函数的 prototype 对象之后，再来看 new 的执行过程：

(1) 创建一个新的对象，并让 this 指针指向它；

(2) 将函数的 prototype 对象的所有成员都赋给这个新对象；

(3) 执行函数体，对这个对象进行初始化操作；

(4) 返回 1 中创建的对象。

与 new 的执行过程相比，这里增加了 prototype 来初始化对象的过程，这也和 prototype 的字面意思相符，即所对应类实例的原型。这个初始化过程发生在函数体(构造器)执行之前，因此可以在函数体内部调用 prototype 中定义的属性和方法。

【实例 6-7】使用构造函数

程序代码如 ex6_7.html 所示。

ex6_7.html

```
<script language="JavaScript" type="text/javascript">
<!--
   // 定义一个只有一个属性 prop 的类
   function class1(){
      this.prop=1;
      this.showProp();
   }
   // 使用函数的 prototype 属性给类定义新成员
   class1.prototype.showProp=function(){
      alert(this.prop);
   }
   // 创建 class1 的一个实例
   var obj1=new class1();
// -->
</script>
```

运行的结果和【实例 6-6】相同。但是和【实例 6-6】代码相比，本实例在 class1 的内部调用了 prototype 中定义的方法 showProp，从而在对象的构造过程中就弹出了对话框，显示 prop 属性的值为 1。

注意：

原型对象的定义必须在创建类实例的语句之前，否则它将不起作用，可参考【实例 6-8】来理解。

【实例 6-8】创建实例后再使用 prototype

程序代码如 ex6_8.html 所示。

ex6_8.html

```
<script language="JavaScript" type="text/javascript">
<!--
   // 定义一个只有一个属性 prop 的类
   function class1(){
      this.prop=1;
      this.showProp();
   }
   // 创建 class1 的一个实例
   var obj1=new class1();
   // 在创建实例的语句之后使用函数的 prototype 属性给类定义新成员，只会对后面创建的对象有效
   class1.prototype.showProp=function(){
      alert(this.prop);
   }
```

```
// -->
</script>
```

这段代码将不会弹出窗口，显示对象并没有执行 showProp 方法，因为该方法的定义在实例化这个类的语句之后。由此可见，prototype 对象专用于设计类的成员，它是和一个类紧密相关的。除此之外，prototype 还有一个重要的属性 constructor，表示对该构造函数的引用。

【实例 6-9】使用 constructor

程序代码如 ex6_9.html 所示。

ex6_9.html

```
<script language="JavaScript" type="text/javascript">
<!--
  function class1(){
    alert(1);
  }
  class1.prototype.constructor(); // 调用类的构造函数
//-->
</script>
```

这段代码运行后将会出现对话框，在上面显示文字“1”，说明一个 prototype 是和一个类的定义紧密相关的。实际上 class1.prototype.constructor 等效于 class1。

4. 一种 JavaScript 类的设计模式

前面已经介绍了如何定义一个类，如何初始化一个类的实例，且类可以在 function 定义的函数体中添加成员，又可以用 prototype 定义类的成员，编写的代码显得混乱。如何以一种更清晰的方式来定义类呢？下面给出了一种类的设计模式。

在 JavaScript 中，由于对象灵活的性质，在构造函数中也可以为类添加成员，在增加灵活性的同时，也增加了代码的复杂度。为了提高代码的可读性和开发效率，可以采用这种定义成员的方式，而使用 prototype 对象来替代，这样 function 的定义就是类的构造函数，符合传统意义上类的实现：类名和构造函数名是相同的。例如：

```
function class1(){ // 构造函数
}
// 成员定义
class1.prototype.someProperty="sample";
class1.prototype.someMethod=function(){ // 方法实现代码
}
```

虽然上面的代码对于类的定义已经清晰了很多，但每定义一个属性或方法，都需要使用一次 class1.prototype，不仅代码变长，且易读性不好。为了使代码变得简洁，可以使用无类型对象的构造方法来指定 prototype 对象，从而实现类的成员定义，程序代码如下：

```
// 定义一个类 class1
function class1(){   // 构造函数
}
// 通过指定 prototype 对象来实现类的成员定义
class1.prototype={
   someProperty:"sample",   someMethod:function(){ // 方法代码
   },
   …// 其他属性和方法
}
```

上面的代码用一种更清晰的方式定义了 class1，构造函数直接用类名来实现，而成员使用无类型对象来定义，以列表的方式实现了所有属性和方法，并且可以在定义的同时初始化属性的值。这更像传统意义上的面向对象语言中类的实现。只是构造函数和类的成员定义被分为了两个部分，这是 JavaScript 中定义类的一种固定模式，使得代码易于理解。

注意：

在一个类的成员之间互相引用，必须通过 this 指针来进行。例如：在上面例子中的 someMethod 方法中，如果要使用属性 someProperty，必须通过 this.someProperty 的形式，因为在 JavaScript 中每个属性和方法都是独立的，它们通过 this 指针联系在一个对象上。

6.4　JavaScript 的浏览器内部对象(DOM)

文档对象(Document Object Model，DOM)是从网页文档里划分出来的对象。在 JavaScript 所涉及的范围内有如下几个“大”对象：window、document、location、navigator、screen 和 history 等。表 6-6 列出了一个文档对象树，其中包含了常用对象。如果要引用其中某个对象，需要将父级的对象都列出来。例如，要引用某表单“applicationForm”的某个文字框“customerName”，需要写成“document.applicationForm.customerName”。

表 6-6　JavaScript 的文档对象树

英 文 名 称	中 文 含 义
navigator	浏览器对象
screen	屏幕对象
window	窗口对象
history	历史对象
location	地址对象
frames[]; Frame	框架对象
document	文档对象
anchors[]; links[]; Link	连接对象
applets[]	Java 小程序对象
embeds[]	插件对象
forms[]; Form	表单对象

(续表)

英 文 名 称	中 文 含 义
Button	按钮对象
Checkbox	复选框对象
elements[]; Element	表单元素对象
Hidden	隐藏对象
Password	密码输入区对象
Radio	单选区域对象
Reset	重置按钮对象
Select	选择区(下拉菜单、列表)
options[]; Option	选择项对象
Submit	提交按钮对象
Text	文本框对象
Textarea	多行文本输入区对象
images[]; Image	图片对象

注意：

JavaScript 是大小写敏感的，表 6-6 中有些对象是全小写的，有些是以大写字母开头的。以大写字母开头的对象表示：引用该对象不使用表中列出的名字，而直接用对象的“名字”(Id 或 Name)，或用它所属的对象数组指定。

6.4.1　浏览器对象 navigator

navigator 对象可以获取用户正在使用的浏览器的版本、用户的浏览器可以控制的 MIME 类型、用户已经安装的插件等。所有这些 navigator 的属性均为只读。该对象包含两个子对象：外挂对象(plugin)和 MIME 类型对象。表 6-7 和表 6-8 分别列出了浏览器对象 navigator 的属性和方法。

表 6-7　浏览器对象 navigator 的属性

属 性 名	说 明
appCodeName	代码
appName	名称
appVersion	版本
language	语言
mimeType	以数组表示所支持的 MIME 类型
platform	编译浏览器的计算机类型
plugins	以数组表示已安装的外挂程序
userAgent	用户代理程序的表头

表 6-8　浏览器对象 navigator 的方法

方　法　名	说　　明
javaEnabled	测试是否支持 Java
plugins.refresh	使新安装的插件有效，并可选重新装入已打开的包含插件的文档
preference	允许一个已标识的脚本获取并设置特定的 navigator 参数
taintEnabled	指定是否允许数据污点

【实例 6-10】对象 navigator 的用法

程序代码如 ex6_10.html 所示。

ex6_10.html

```
<Script>
  with (document) {
    write ("你的浏览器信息：<OL>");
    write ("<LI>代码："+navigator.appCodeName);
    write ("<LI>名称："+navigator.appName);
    write ("<LI>版本："+navigator.appVersion);
    write ("<LI>语言："+navigator.language);
    write ("<LI>编译平台："+navigator.platform);
    write ("<LI>用户表头："+navigator.userAgent);
}
</Script>
```

本实例利用 document 对象显示浏览器的信息，运行后的显示结果如图 6-3 所示。

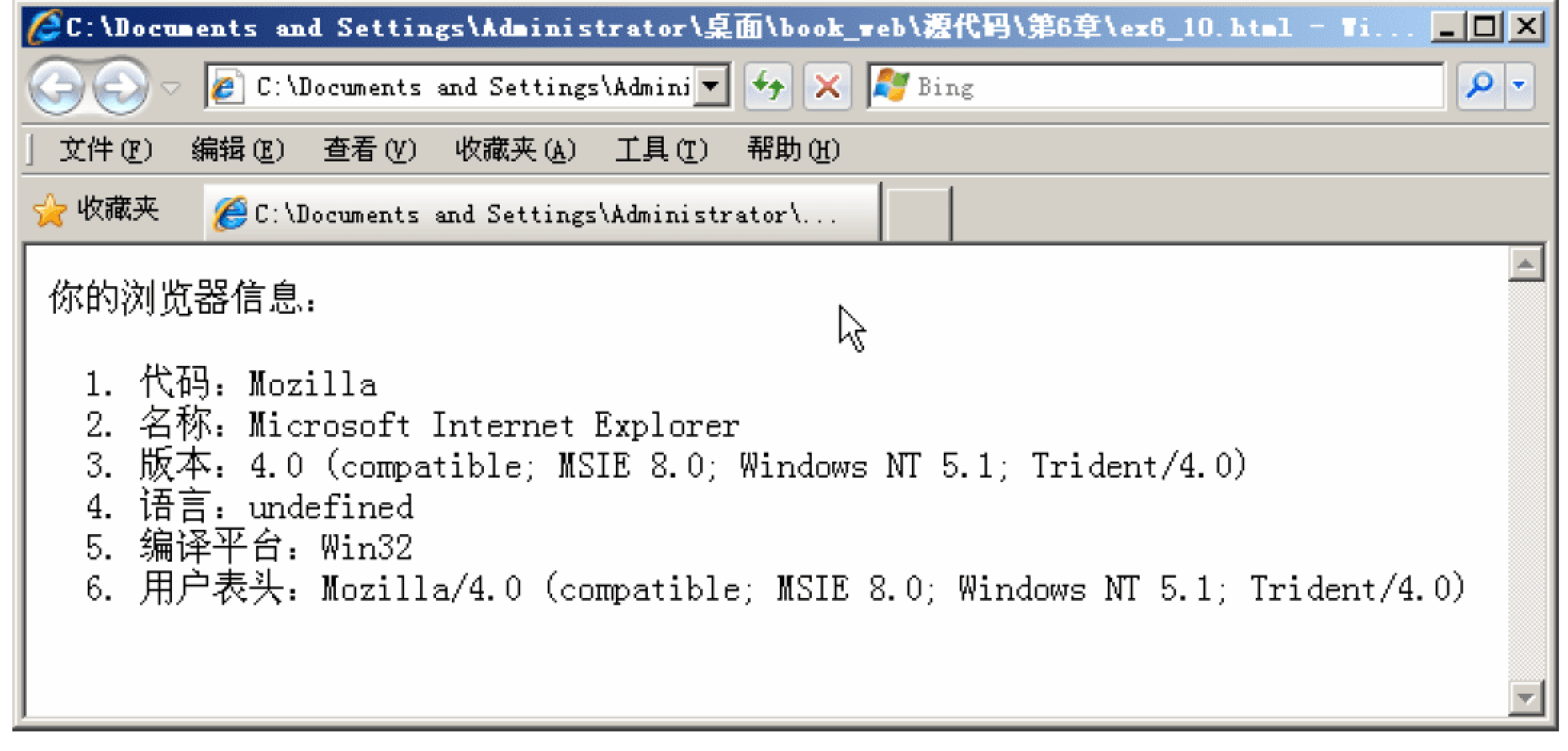


图 6-3　对象 navigator 应用示例

1. 外挂对象(navigator.plugin)

Plugin 对象是一个安装在客户端的插件。所谓插件，就是浏览器用于显示特定类型嵌入数据时调用的软件模块。现在访问某些网站就会自动安装插件，这个问题困扰了很多用户。实际上每个 Plugin 对象本身都是一个数组，其中包含的每个元素分别对应于每个该插件支持的 MIME 类型。而数组的每个元素都是一个 MimeType 对象。外挂对象(navigator.plugin)的属性如表 6-9 所示。

表 6-9　外挂对象(navigator.plugin)的属性

属 性 名	说 明
description	外挂程序模块的描述
filename	外挂程序模块的文件名
length	外挂程序模块的个数
name	外挂程序模块的名称

【实例 6-11】列出所有外挂对象(navigator.plugin)

程序代码如 ex6_11.html 所示。

ex6_11.html

```
<script language="JavaScript" type="text/javascript">
<!--
  var len = navigator.plugins.length;
  with (document) {
    write ("你的浏览器共支持" + len + "种 plug-in：<BR>");
    write ("<TABLE BORDER>")
    write ("<CAPTION>PLUG-IN 清单</CAPTION>")
    write ("<TR><TH> <TH>名称<TH>描述<TH>文件名")
    for (var i=0; i<len; i++)
      write("<TR><TD>" + i +
        "<TD>" + navigator.plugins[i].name +
        "<TD>" + navigator.plugins[i].description +
        "<TD>" + navigator.plugins[i].filename);
}
//-->
</script>
```

本实例利用 navigator.plugin 对象显示了浏览器外挂程序的有关信息。其中的关键是通过循环来遍历客户端浏览器所有已安装的插件，运行的结果如图 6-4 所示。

Mozilla Firefox

文件(F) 编辑(E) 查看(V) 转到(G) 书签(B) 工具(T) 帮助(H)

file:///E:/0UNfinished/web_bk/ex/ex5_11.html 转到

入门须知 最新头条

你的浏览器共支持16种plug-in:

PLUG-IN 清单

	名称	描述	文件名
0	Shockwave Flash	Shockwave Flash 9.0 r16	NPSWF32.dll
1	Mozilla Default Plug-in	Default Plug-in	npnul32.dll
2	RealJukebox NS Plugin	RealJukebox Netscape Plugin	nprjplug.dll
3	RealPlayer(tm) G2 LiveConnect-Enabled Plug-In (32-bit)	RealPlayer(tm) LiveConnect-Enabled Plug-In	nppl3260.dll
4	RealPlayer Version Plugin	6.0.12.1739	nprpjplug.dll
5	Java(TM) 2 Platform Standard Edition 5.0 Update 10	Java Plug-in 1.5.0_10 for Netscape Navigator (DLL Helper)	NPJava11.dll
6	Java(TM) 2 Platform Standard Edition 5.0 Update 10	Java Plug-in 1.5.0_10 for Netscape Navigator (DLL Helper)	NPJava12.dll
7	Java(TM) 2 Platform Standard Edition 5.0 Update 10	Java Plug-in 1.5.0_10 for Netscape Navigator (DLL Helper)	NPJava13.dll
8	Java(TM) 2 Platform Standard Edition 5.0 Update 10	Java Plug-in 1.5.0_10 for Netscape Navigator (DLL Helper)	NPJava14.dll
9	Java(TM) 2 Platform Standard Edition 5.0 Update 10	Java Plug-in 1.5.0_10 for Netscape Navigator (DLL Helper)	NPJava32.dll
10	Java(TM) 2 Platform Standard Edition 5.0 Update 10	Java Plug-in 1.5.0_10 for Netscape Navigator (DLL Helper)	NPJPI150_10.dll
11	Java(TM) 2 Platform Standard Edition	Java Plug-in 1.5.0_10 for Netscape	NPOJI610.dll

完成

图 6-4　外挂对象(navigator.plugin)应用示例

2. 对象 navigator.mimeTypes

这是客户端所支持的 MIME(Multipart Internet Mail Extension，即：多部分网际邮件扩展) 类型，mimeType 对象的属性如表 6-10 所示。

表 6-10　navigator.mimeTypes 的属性

属　性　名	说　　明
description	MIME 类型的描述
enablePlugin	对应到哪个外挂模块
length	MIME 类型的数目
suffixes	MIME 类型的扩展名
type	MIME 类型的名称

通常无须自行创建 mimeTypes 对象。这些对象是预先定义的 JavaScript 对象，可通过 navigator 或 plugin 对象的 mimeTypes 数组来访问这些对象，每个 mimeTypes 对象都是 mimeTypes 数组中的一个元素，其用法如下：

```
navigator.mimeTypes[index]
```

这里 index 或者是表明由客户端支持的 MIME 类型的整型值，也可以是包含了 mimeTypes 对象类型(来自于 mimeTypes.type 属性)的字符串。

【实例 6-12】对象 navigator.mimeTypes 应用示例

程序代码如 ex6_12.html 所示。

ex6_12.html

```
<script language="JavaScript" type="text/javascript">
<!--
  var len = navigator.mimeTypes.length;
  with (document) {
    write ("你的浏览器共支持" + len + "种 MIME 类型：");
    write ("<TABLE BORDER>")
    write ("<CAPTION>MIME type  清单</CAPTION>")
    write ("<TR><TH> <TH>名称<TH>描述<TH>扩展名<TH>附注")
    for (var i=0; i<len; i++) {
      write("<TR><TD>" + i +
        "<TD>" + navigator.mimeTypes[i].type +
        "<TD>" + navigator.mimeTypes[i].description +
        "<TD>" + navigator.mimeTypes[i].suffixes + "<TD>" +
        navigator.mimeTypes[i].enabledPlugin.name);
    }
  }
//-->
</script>
```

本实例利用 navigator.mimeTypes 对象显示了浏览器所支持的 MIME 类型信息。其中

使用了循环来遍历客户端浏览器的所有已安装的插件，运行结果如图 6-5 所示。

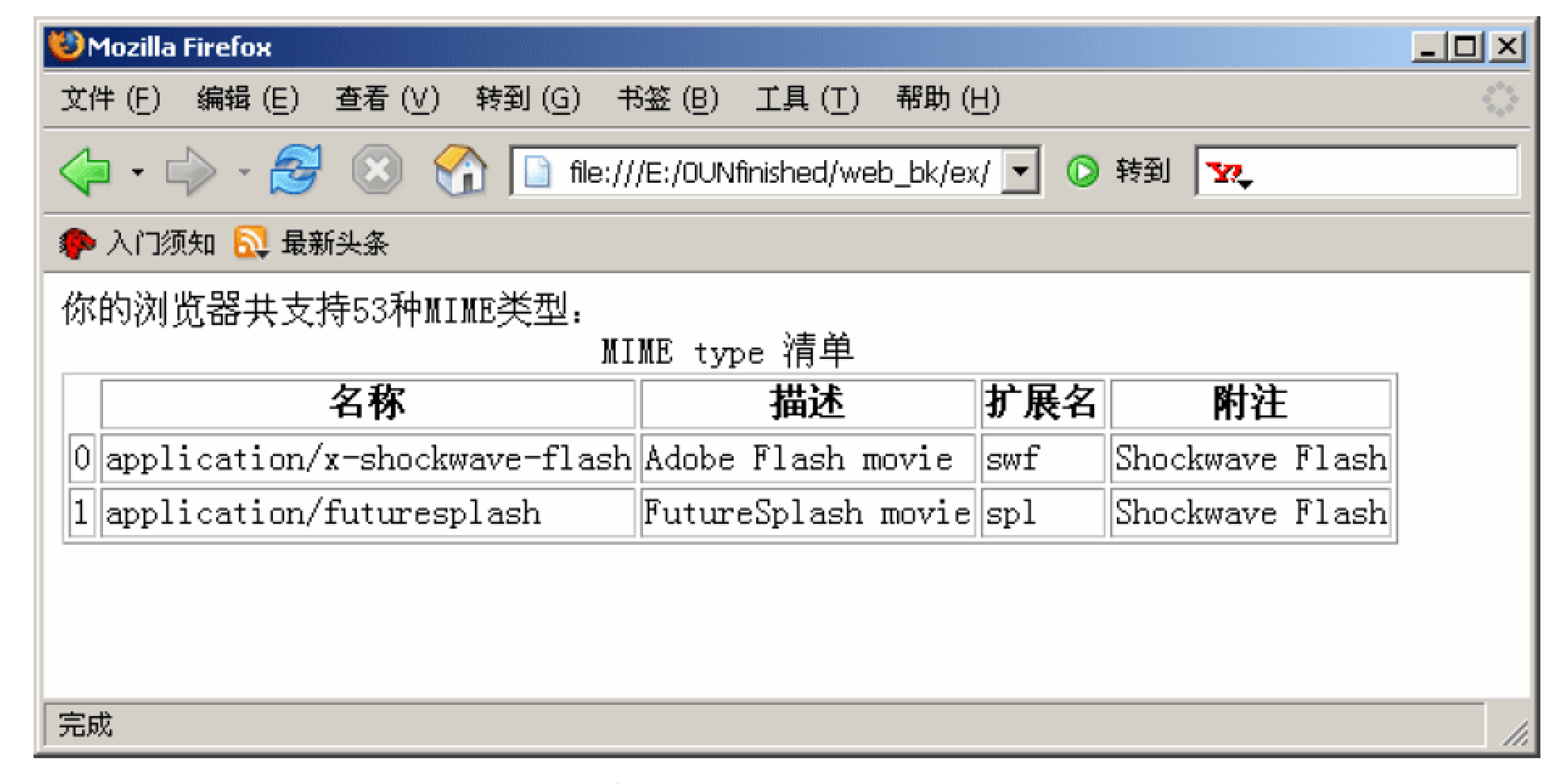


图 6-5　外挂对象(navigator. mimeTypes)应用示例

6.4.2　窗口对象 window

窗口对象 window 的属性如表 6-11 所示。

表 6-11　窗口对象 window 的属性

属　性　名	说　　明
document	当前文件的信息
location	当前 URL 的信息
name	窗口名称
status	状态栏的临时信息
defaultStatus	状态栏默认信息
history	该窗口最近查阅过的网页
closed	判断窗口是否关闭，返回布尔值
opner	open 方法打开窗口的源窗口
outerHeight	窗口边界的垂直尺寸，px
outerWidth	窗口边界的水平尺寸，px
pageXOffset	网页 x-position 的位置
pageYOffset	网页 y-position 的位置
innerHeight	窗口内容区的垂直尺寸，px
innerWidth	窗口内容区的水平尺寸，px
screenX	窗口距屏幕左边界的像素
screenY	窗口距屏幕上边界的像素
self	当前窗口
top	最上方的窗口
parent	当前窗口或框架的框架组
frames	对应到窗口中的框架
length	框架的个数
locationbar	浏览器地址栏

(续表)

属　性　名	说　　明
menubar	浏览器菜单栏
scrollbars	浏览器滚动条
statusbar	浏览器状态栏
toolbar	浏览器工具栏
offscreenBuffering	是否更新窗口外的区域
personalbars	浏览器的工具栏，仅针对 navigator

窗口对象 window 的事件及方法如表 6-12 所示。

表 6-12　窗口对象 window 的事件及方法

属　性　名	说　　明
alert(信息字串)	弹出警告信息
confirm(信息字串)	显示确认信息对话框
prompt(提示字串[，默认值])	显示提示信息，并提供可输入的字段
atob(译码字串)	对 base-64 编码字串进行译码
btoa(字串)	将进行 base-64 编码
back()	回到历史记录的上一网页
forward()	加载历史记录中的下一网页
open(URL，窗口名称[，窗口规格])	打开窗口
focus()	焦点移到该窗口
blur()	窗口转成背景
stop()	停止加载网页
close()	关闭窗口
enableExternalCapture()	允许有框架的窗口获取事件
disableExternalCapture()	关闭捕捉外部标志
captureEvents(事件类型)	捕捉窗口的特定事件
routeEvent(事件)	传送已捕捉的事件
handleEvent(事件)	使特定事件的处理生效
releaseEvents(事件类型)	释放已获取的事件
moveBy(水平点数，垂直点数)	相对定位
moveTo(x 坐标，y 坐标)	绝对定位
setResizable(true|false)	是否允许调整窗口大小
resizeBy(水平点数，垂直点数)	相对调整窗口大小
resizeTo(宽度，高度)	绝对调整窗口大小
scroll(x 坐标，y 坐标)	绝对滚动窗口
scrollBy(水平点数，垂直点数)	相对滚动窗口
scrollTo(x 坐标，y 坐标)	绝对滚动窗口
setInterval(表达式，毫秒)	设置间隔

(续表)

属　性　名	说　　明
setTimeout(表达式，毫秒)	设置超时
clearInterval(定时器对象)	清除定时器
clearTimeout(定时器对象)	清除超时
home()	进入浏览器设置的主页
find([字串[,caseSensitivr,backward]])	查找窗口中特定的字串
print()	打印
setHotKeys(true|false)	激活或关闭组合键
setZOptions()	设置窗口重叠时的堆栈顺序

窗口对象 window 的事件包括如下。

- onBlur：失去焦点事件；
- onDragDrop：拖放事件；
- onError：出错事件；
- onFocus：获得焦点事件；
- onLoad：载入事件；
- onMove：移动事件；
- onResize：改变大小事件；
- onUnload：对象销毁事件。

【实例 6-13】检查输入框

程序代码如 ex6_13.html 所示。

ex6_13.html

```
<HTML>
<HEAD>
<script language="JavaScript" type="text/javascript">
<!--
    function checkPassword(testObject) {
        if (testObject.value.length < 4) {
            alert("密码长度不得小于 4");
            testObject.focus();
            testObject.select();
        }
    }
//-->
</script>
</HEAD>
<BODY>
    <INPUT TYPE="text" onBlur="checkPassword(this)">
</BODY>
```

本实例利用 window 对象响应 onblur 事件，来检查输入框中输入字符的个数。若输入的字符数小于 4 个，则当输入框失去焦点时就弹出一个对话框，显示“密码长度不得小于 4”。本实例中不仅调用 alert()方法弹出对话框，还调用了文本框对象的 focus()和 select()方法。

【实例 6-14】简易时钟

程序代码如 ex6_14.html 所示。

ex6_14.html

```
<HTML>
  <HEAD>
    <SCRIPT LANGUAGE="JavaScript">
    <!--
      var timerID = null
      var timerRunning = false
      function stopclock(){
          if(timerRunning)
              clearTimeout(timerID)
          timerRunning = false
      }
      function startclock(){
          // 确保时钟处于停止状态
          stopclock()
          showtime()
      }
      function showtime(){
          var now = new Date()
          var hours = now.getHours()
          var minutes = now.getMinutes()
          var seconds = now.getSeconds()
          var timeValue = "" + ((hours > 12) ? hours - 12 : hours)
          timeValue += ((minutes < 10) ? ":0" : ":") + minutes
          timeValue += ((seconds < 10) ? ":0" : ":") + seconds
          timeValue += (hours >= 12) ? " P.M." : " A.M."
          document.clock.face.value = timeValue
          timerID = setTimeout("showtime()",1000)
          timerRunning = true
      }
    //-->
    </SCRIPT>
  </HEAD>
  <BODY onLoad="startclock()">
    <FORM NAME="clock" onSubmit="0">
        <INPUT TYPE="text" NAME="face" SIZE=12 VALUE ="">
    </FORM>
  </BODY>
</HTML>
```

这个例子利用 BODY 元素的 onLoad 事件启动 startclock()，在此函数中首先调用

stopclock()方法停止时钟的运行，再调用 showtime 来显示时间。其中.setTimeout、.clearTimeout 分别为启动和停止某个按时间间隔来运行的方法。此外，本实例还调用了 Date 等对象方法。读者可自行运行该实例来查看运行的效果。

【实例 6-15】屏幕自动滚动

程序代码如 ex6_15.html 所示。

ex6_15.html

```
<HTML>
  <HEAD>
    <Script>
    <!--
      function scrollIt() {
        for (y=1; y<=2000; y++) {
          scrollTo(1,y);
        }
      }
    //-->
    </Script>
  </HEAD>
  <Body onDblClick=scrollIt()>
    双击鼠标，画面会自动滚动...
    <br><br><br><br><br><br><br><br><br><br><br><br><br><br>
    <br><br><br><br><br><br><br><br><br><br><br><br><br><br>
    <br><br><br><br><br><br><br><br><br><br><br><br><br><br>
    <br><br><br><br><br><br><br><br><br><br><br><br><br><br>
    ... The End ...
  </Body>
</HTML>
```

这个例子利用 BODY 元素的 onDblClick 事件启动 scrollIt()方法，在此函数中首先调用 window 对象的 scrollTo()方法来实现窗口的自动滚动。因此，在此浏览器文档显示界面上任何一个地方用鼠标双击，就会出现屏幕自动滚动的效果。

【实例 6-16】弹出新窗口

网页中常见到弹出新窗口的网站，它是利用 window 对象的 open()方法实现的，程序代码如 ex6_16.html 所示。

ex6_16.html

```
<HTML>
 <HEAD><TITLE>欢迎光临</TITLE>
    <Script>
      document.write ("这是一个测试窗口打开和关闭的 JavaScript 代码");
      open('ex6_13.html','','height=100,width=300');
    </Script>
  </HEAD>
  <BODY onClick="self.close()">
```

```
    <CENTER><FONT COLOR="blue" SIZE="5">欢迎光临</FONT><BR>
      单击窗口中任何地方，可以关闭本窗口<BR>
    </CENTER>
  </BODY>
</HTML>
```

本实例利用 BODY 元素的 onClick 事件启动 self.close()方法来实现窗口的关闭。前面的 JavaScript 代码使得在窗口打开时自动调用 open('ex6_13.html','', 'height=100 ,width=300');，这个函数可以按照设置的要求打开一个浏览器窗口。

注意：

如果在某些允许设置“不允许弹出窗口”的浏览器中进行了设置，就无法弹出窗口了。

调用 open()函数的基本格式为：

```
[var 新窗口对象名=]window.open("url","windowName","windowFeature")
```

其中的 url 是新窗口的地址，windowName 为新窗口的名称，而 windowFeature 中可以设置新窗口的一些属性，允许设置的参数如表 6-13 所示。

表 6-13　open()函数 windowFeature 参数的设置

名　称	含　义
alwaysLowered	是否将窗口显示的堆栈后推一层
alwaysRaised	是否将窗口显示的堆栈上推一层
dependent	是否将该窗口与当前窗口产生依存关系
fullscreen	是否满屏显示
directories	是否显示连接工具栏
location	是否显示网址工具栏
menubar	是否显示菜单工具栏
scrollbars	是否显示滚动条
status	是否显示状态栏
titlebar	是否显示标题栏
toolbar	是否显示标准工具栏
resizable	是否可以改变窗口的大小
screenX	窗口左边界距离
screenY	窗口上边界距离
top	窗口的上边界
width	窗口的宽度
height	窗口的高度
left	窗口的左边界
outerHeight	窗口外边界的高度
personalbar	是否显示个人工具栏

【实例 6-17】弹出新窗口参数演示

程序代码如 ex6_17.html 所示。

ex6_17.html

```
<HTML>
  <HEAD><TITLE>window.open 参数示例</TITLE>
  <BODY >
    <FORM NAME="myform">
      <INPUT TYPE="button" NAME="Button1" VALUE="打开新窗口!"
        onClick="window.open ('ex6_15.html', 'newWindow',
        'scrollbars=no,status=no,width=300,height=300')">
    </FORM>
  </BODY>
</HTML>
```

本实例运行后在浏览器中出现一个“打开新窗口!”的按钮，单击后可弹出一个新窗口，由于设置了“scrollbars=no,status=no,width=300,height=300”，因此，弹出的新窗口没有滚动条，也没有状态栏，窗口的宽度和高度均为 300。请读者自行运行，查看实际结果。

6.4.3　屏幕对象 screen

screen 对象描述了屏幕的显示及颜色属性。该对象包含了允许获取的关于用户显示情况信息的只读属性。表 6-14 列出了 screen 对象的属性以及说明。

表 6-14　屏幕对象 screen 的属性

属性名称	说明
availHeight	屏幕区域的可用高度
availWidth	屏幕区域的可用宽度
colorDepth	颜色深度(8btit/16bit/24bit/32bit)
height	屏幕区域的实际高度
width	屏幕区域的实际宽度
location	是否显示网址工具栏

【实例 6-18】屏幕对象 screen 的使用

程序代码如 ex6_18.html 所示。

ex6_18.html

```
<script language="JavaScript" type="text/javascript">
<!--
  with (document) {
    write ("您的屏幕显示设定值如下：<P>");
    write ("屏幕的实际高度为", screen.availHeight, "<BR>");
    write ("屏幕的实际宽度为", screen.availWidth, "<BR>");
    write ("屏幕的色盘深度为", screen.colorDepth, "<BR>");
```

```
        write ("屏幕区域的高度为", screen.height, "<BR>");
        write ("屏幕区域的宽度为", screen.width);
    }
    //-->
    </script>
```

本实例运行后将当前屏幕设置的有关参数显示出来，网页开发人员可以利用这个对象来获取客户端的设置，进而控制网页以恰当的方式显示，本实例的运行结果如图 6-6 所示。

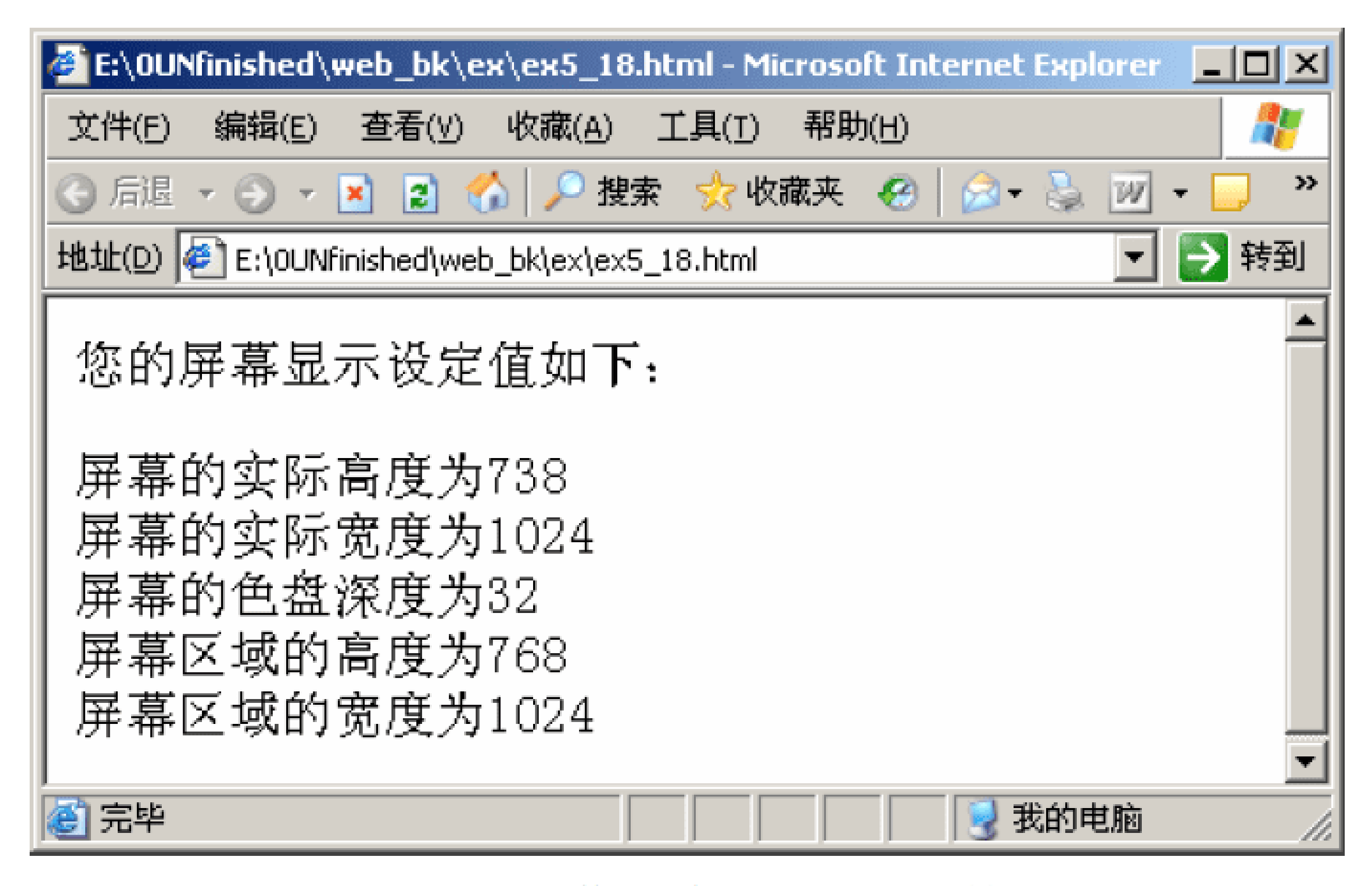


图 6-6　屏幕对象 screen 的使用

6.4.4　事件对象 event

当事件发生时，浏览器自动建立该对象，其中可能包含该事件的类型、鼠标坐标参数等。表 6-15 列出了 event 对象的属性以及说明。

表 6-15　事件对象 event 的属性

属 性 名 称	说　明
data	返回拖曳对象的 URL 字符串(dragDrop)
width	该窗口或框架的宽度
height	该窗口或框架的高度
pageX	光标相对于该网页的水平位置
pageY	光标相对于该网页的垂直位置
screenX	光标相对于该屏幕的水平位置
screenY	光标相对于该屏幕的垂直位置
target	该事件被传送到的对象
type	事件的类型
which	数值表示的键盘或鼠标键：1/2/3(左键/中键/右键)
layerX	光标相对于事件发生层的水平位置
layerY	光标相对于事件发生层的垂直位置
x	相当于 layerX
y	相当于 layerY

【实例 6-19】显示鼠标位置的新窗口

程序代码如 ex6_19.html 所示。

ex6_19.html

```
<script language="JavaScript" type="text/javascript">
<!--
   function getEvent(evnt) {
   eventWin = open ('','','width=200,height=100');
      with (eventWin.document) {
         write("事件类型： ", event.type);
         write("<br>鼠标的 x 坐标： ", event.screenX);
         write("<br>鼠标的 y 坐标： ", event.screenY);
      }
   }
   document.write ("单击...")
   document.onmousedown = getEvent;
//-->
</script>
```

本实例运行后如果在浏览器显示区域中单击鼠标，将出现一个新窗口，其中显示事件对象 event 的有关参数，显示的内容为事件类型以及发生该事件时光标的位置。由于此处仅捕捉了鼠标键按下的事件，因此仅对于该事件有效，其他的动作不产生任何显示，运行结果如图 6-7 所示。

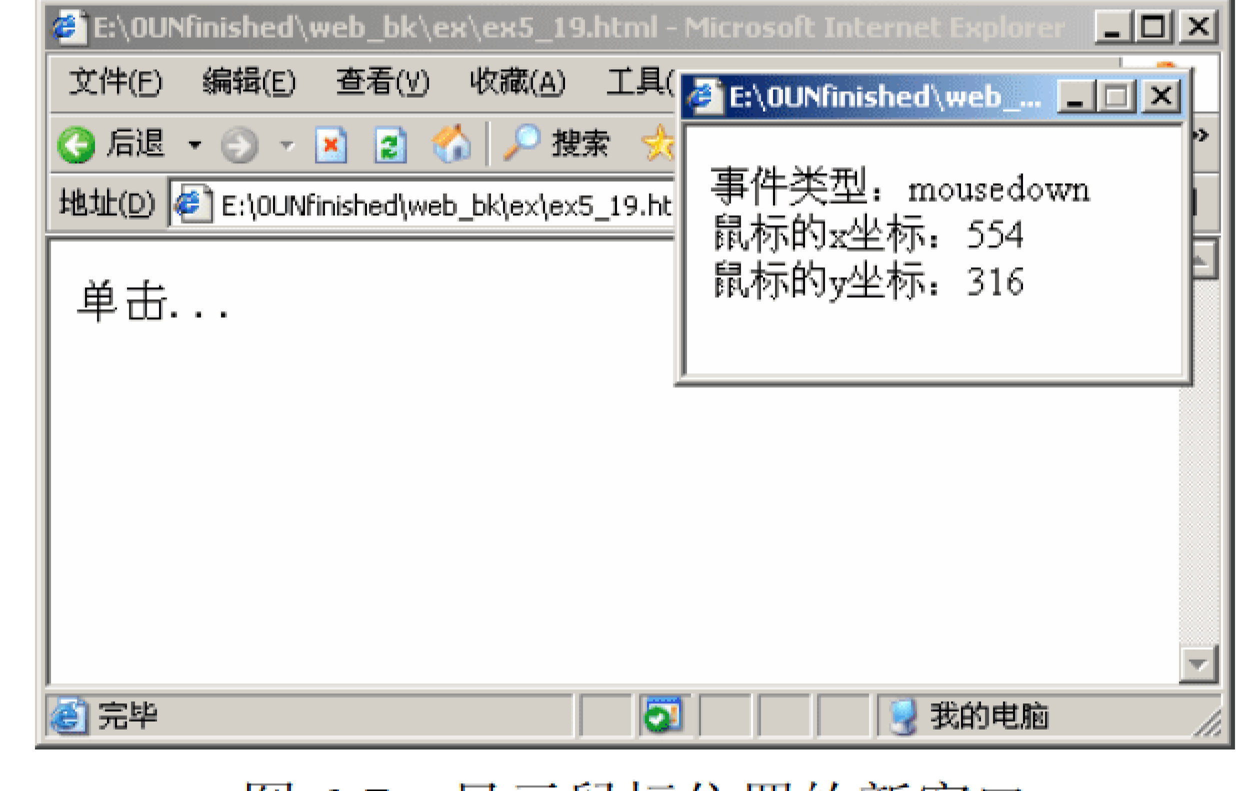


图 6-7　显示鼠标位置的新窗口

【实例 6-20】事件对象 event 的使用

程序代码如 ex6_20.html 所示。

ex6_20.html

```
<script language="JavaScript" type="text/javascript">
<!--
   function getCoordinate(evnt) {
      if (document.all) {
         x = event.screenX;
         y = event.screenY;
      }
      else {
         x = evnt.screenX;
         y = evnt.screenY;
      }
      status = "水平坐标： "+ x + "； 垂直坐标： "+ y;
   }
   function whichKey(evnt) {
      if (document.all) {
```

```
            x = event.button;
            if( x==1 ) alert("你单击了左键");
            if( x==2 ) alert("你单击了右键");
        }
        else {
            x = evnt.button;
            if( x==1 ) alert("你单击了左键");
            if( x==3 ) alert("你单击了中键");
            return false;
        }
    }
    document.onmousedown = whichKey;
    document.onmousemove = getCoordinate;
    document.write("请单击鼠标左/右键，并注意状态栏的显示！");
//-->
</script>
```

本实例运行后在浏览器的状态栏显示了鼠标的位置，当鼠标在显示区域中的任何地方单击时，将显示所单击的是左键或右键的对话框，状态栏实时显示光标当前的位置，实现该功能利用了 onmousedown 和 onmousemove 两个事件，运行结果如图 6-8 所示。

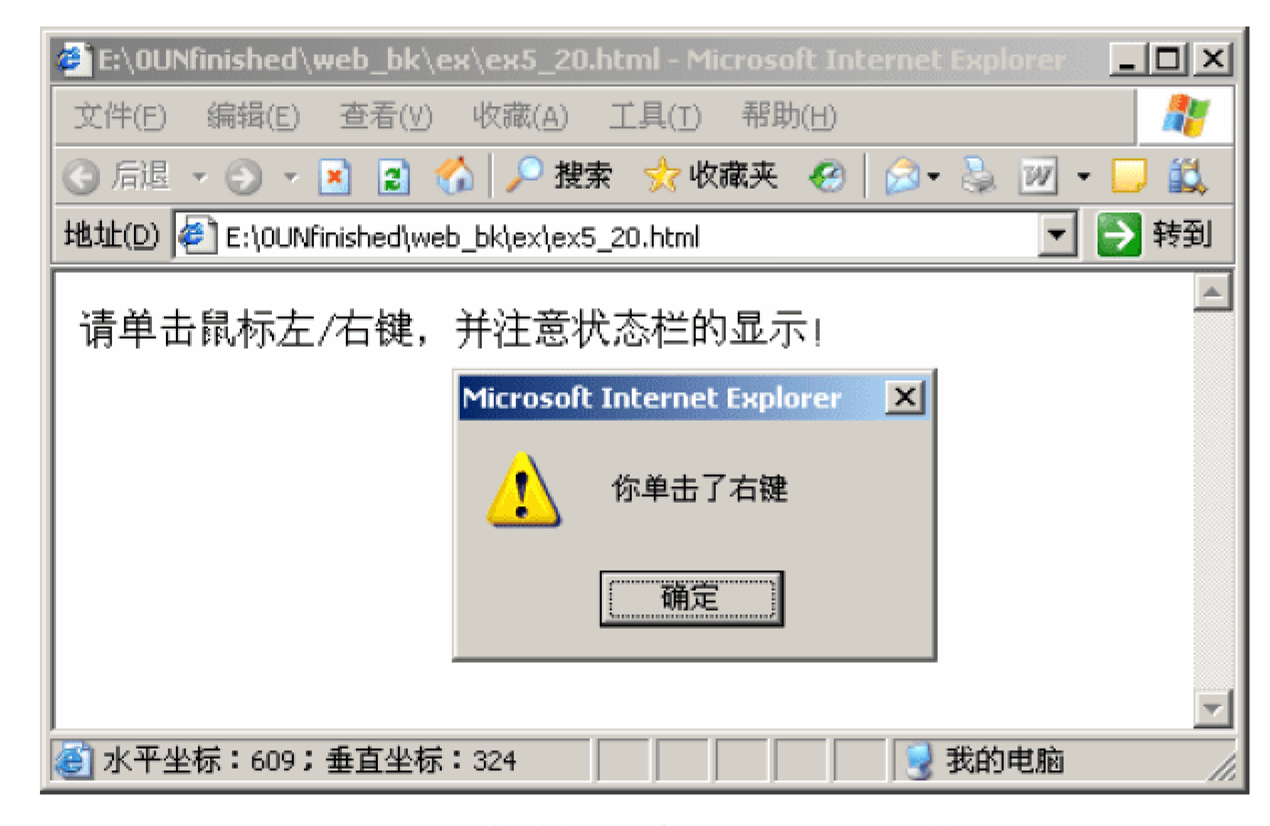


图 6-8　事件对象 event 的使用

6.4.5　历史对象 history

常见的浏览器中可以保存历史访问记录(客户端最近访问的网址清单)，history 对象实现这个功能，其常用属性和方法如表 6-16 及表 6-17 所示。可以通过 history[i]获得历史记录列表中的第 i 条历史记录。

表 6-16　历史对象 history 的属性

属 性 名 称	说　　明
current	当前历史记录的网址
length	存储在记录清单中的网址数目
next	下一个历史记录的网址
previous	上一个历史记录的网址

表 6-17　历史对象 history 的方法

方 法 名 称	说　　明
back()	回到上一个历史记录中的网址
forward()	回到下一个历史记录中的网址
go(整数或 URL)	前往历史记录中的网址

其中，go(-1)则代表载入前一条历史记录，它和 back()方法的功能相同；go(1)代表载入后一条历史记录，它和 forward()方法的功能相同。

【实例 6-21】历史对象 history 的使用

程序代码如 ex6_21.html 所示。

ex6_21.html

```
<HTML>
  <HEAD>
    <TITLE>history 对象的使用</TITLE>
    <SCRIPT LANGUAGE="javascript">
    <!--
      function goback(){
        history.back();
      }
      function goforward(){
        history.forward();
      }
    //-->
    </SCRIPT>
  </HEAD>
  <BODY>
    <FORM>
      <INPUT TYPE="button" VALUE="后退" OnClick="goback()">
      <INPUT TYPE="button" VALUE="前进" OnClick="goforward()">
    </FORM>
  </BODY>
</HTML>
```

本实例运行后在浏览器中出现了两个按钮“前进”和“后退”，其功能与浏览器上所提供的功能一样，不同的是它显示于浏览器的显示区域，且可以通过程序进行控制。

6.4.6　位置对象 location

位置对象 location 用来表示特定窗口的 URL 信息。它描述了与一个给定的 window 对象相关联的完整 URL，它的每个属性都描述了 URL 的不同特性。位置对象 location 常用的属性如表 6-18 所示。

表 6-18　位置对象 location 的属性

属 性 名 称	说　明
hash	锚点名称
host	主机名称
hostname	host:port
href	完整的 URL 字符串
pathname	路径
port	端口

(续表)

属 性 名 称	说　明
protocol	协议
search	查询信息

通常情况下，一个 URL 会有下面的格式：

```
协议//主机:端口/路径名称#哈希标识?搜索条件
```

例如，http://home.netscape.com/assist/extensions.html#topic1?x=7&y=2，它满足下列要求。

- 协议：URL 的起始部分，直到包含到第一个冒号。
- 主机：描述了主机和域名，或者一个网络主机的 IP 地址；它可以是以下几种形式：view-source: 显示源代码、http:、file:、ftp:、mailto:、news:和 gopher:等。
- 端口：描述了服务器用于通信的通信端口。
- 路径名称：描述了 URL 的路径方面的信息。
- 哈希标识：描述了 URL 中的锚名称，包括哈希掩码(#)。此属性只应用于 HTTP 的 URL。
- 搜索条件：描述了该 URL 中的任何查询信息，包括问号。此属性只应用于 HTTP 的 URL。“搜索条件”字符串包含变量和值的配对；每对之间由一个“&”连接。

位置对象的方法主要包括 2 个：reload()表示重新加载；replace(网址)用于指定的网页取代当前网页。

【实例 6-22】位置对象 location 的使用

程序代码如 ex6_22.html 所示。

ex6_22.html

```
<HTML>
  <HEAD>
    <Script>
    <!--
      var sec = 5;
      function countDown() {
        if (sec > 0) {
          num.innerHTML = sec--;
        }
        else
          location = "http://www.csai.com.cn/";
        }
    //-->
    </Script>
  </HEAD>
  <BODY onLoad="setInterval('countDown()', 1000)">
    <CENTER>
      希赛网
```

```
        <H2>http://www.csai.com.cn/</H2>
          五秒钟后自动带你前往<BR>
        <FONT ID="num" SIZE="6">开始倒计数</FONT>
      </BODY>
    </HTML>
```

本实例运行后在浏览器中出现了一个倒计时的秒数显示，当其显示为 1 之后执行转向功能，该功能利用了位置对象 location 实现。

6.4.7　文件对象 document

文件对象 document 代表了当前 HTML 对象，它是由<BODY>标签组构成的，对每个 HTML 文件会自动建立一个文件对象。它支持的事件包括：onClick、onDbClick、onKeyDown、onKeyPress、onKeyUp、onMouseDown 和 onMouseOver 等。表 6-19 和表 6-20 分别列出了文件对象 document 的属性和方法。

表 6-19　文件对象 document 的属性

属 性 名 称	说　　明
linkColor	设置超链接的颜色
alinkColor	作用中的超链接的颜色
vlinkColor	链接的超链接颜色
links	以数组索引值表示所有超链接
URL	该文件的网址
anchors	以数组索引值表示所有锚点
bgColor	背景颜色
fgColor	前景颜色
classes	文件中的 class 属性
cookie	设置 cookie
domain	指定服务器的域名
formName	以表单名称表示所有表单
forms	以数组索引值表示所有表单
images	以数组索引值表示所有图像
layers	以数组索引值表示所有 layer
embeds	文件中的 plug-in
applets	以数组索引值表示所有 applet
plugins	以数组索引值表示所有插件程序
referrer	代表当前打开文件的网页的网址
tags	指出 HTML 标签的样式
title	该文档的标题
width	该文件的宽度(px)
lastModified	文件最后修改时间

表 6-20　文件对象 document 的方法

方 法 名 称	说　　明
captureEvents(事件)	设置要获取指定的事件
close()	关闭输出字符流，强制显示数据内容
getSelection()	取得当前选取的字串
handleEvent(事件)	使事件处理器生效
open([mimeType,[replace]])	打开字符流
releaseEvents(事件类型)	释放已获取的事件
routeEvent(事件)	传送已捕捉的事件
write(字串)	写字串或数值到文件中
writeln(字串)	分行写字串或数值到文件中(<pre>..</pre>)
cookie	设置 cookie
domain	指定服务器的域名
formName	以表单名称表示所有表单
forms	以数组索引值表示所有表单
images	以数组索引值表示所有图像
layers	以数组索引值表示所有 layer

【实例 6-23】文件对象 document 示例

程序代码如 ex6_23.html 所示。

ex6_23.html

```
<HTML>
   <HEAD>
      <Script>
      <!--
         document.bgColor = "gray";
         document.fgColor = "black";
         document.linkColor = "red";
         document.alinkColor = "blue";
         document.vlinkColor = "purple";
         var update_date = document.lastModified;
         var formated_date = update_date.substring(0,10);
         document.write("本网页更新日期：" + update_date + "<BR>")
         document.write("本网页更新日期：" + formated_date+ "<BR>")
      //-->
      </Script>
   </HEAD>
   <BODY>
      <BR>测试文件对象的颜色属性：<BR>
      <A HREF="http://www.csai.com.cn">希赛网</A>
   </BODY>
</HTML>
```

本实例利用了 document 的有关颜色设置方面的一些属性来控制网页的色彩，通过这

个方式有效地修改了浏览器中对于链接在不同状态下所显示颜色的默认设置。另外，利用 lastModified 这个属性可以显示网页最后更新的时间，运行后的显示如图 6-9 所示。

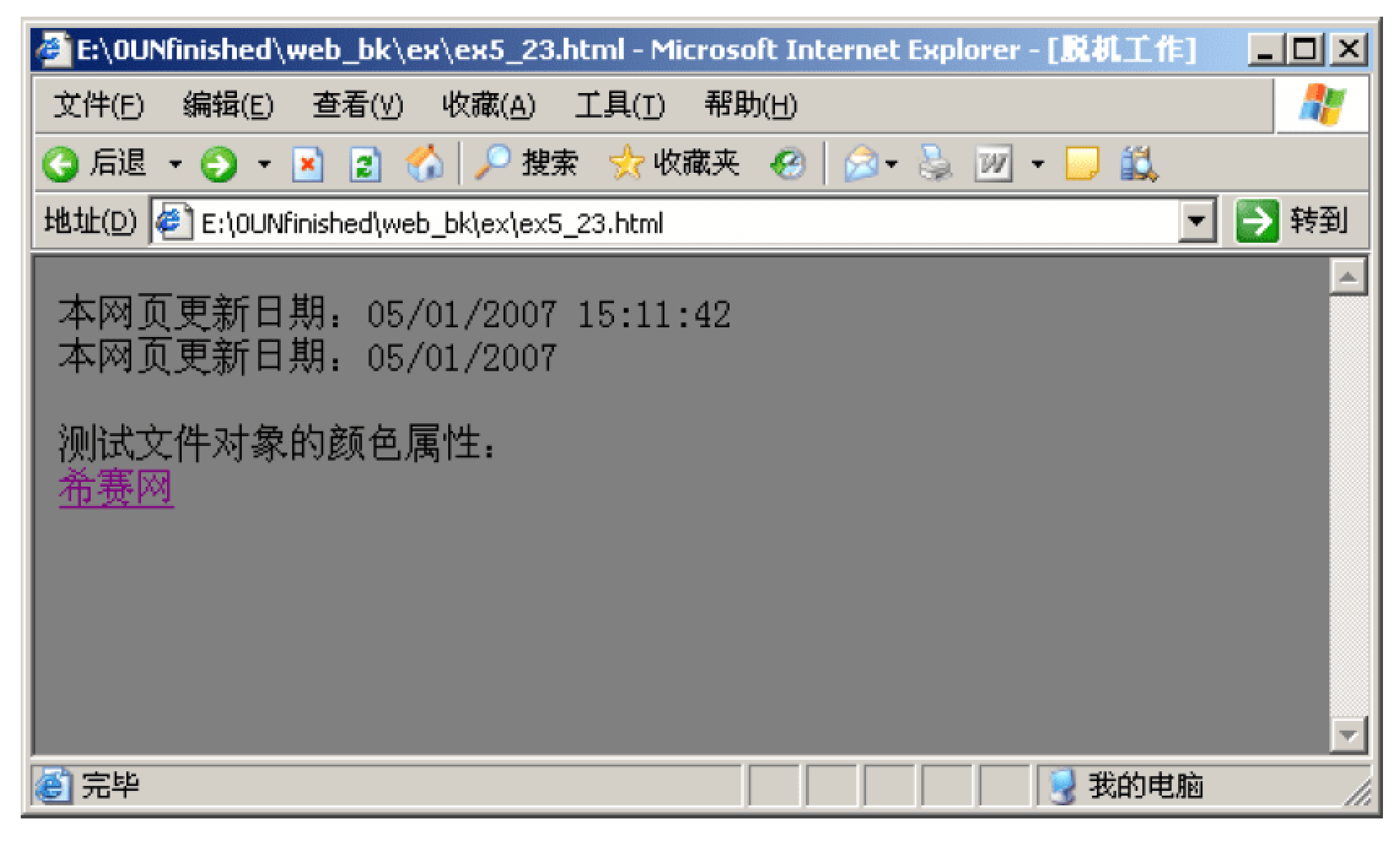


图 6-9　文件对象 document 示例

6.4.8　链接对象 Link

链接对象 Link 是文件对象 document 的子对象，通常网页中的链接会被自动看作链接对象，并按照顺序，可分别表示为 document.links[0]、document.links[1]等，链接对象 Link 的属性如表 6-21 所示。

表 6-21　链接对象 Link 的属性

属 性 名 称	说　　明
hash	URL 中的锚点名称
host	主机域名或 IP 地址
hostname	当前 URL 中的主机名
href	完整的 URL 字串
pathname	URL 中 path 部分
port	URL 中端口部分
protocol	URL 中通信协议部分
search	URL 中查询字串部分
target	代表目标的窗口
text	表示 A 标签中的文字
x	链接对象的左边界
y	链接对象的右边界

【实例 6-24】链接对象 Link 的使用

程序代码如 ex6_24.html 所示。

ex6_24.html

```
<HTML>
  <HEAD>
```

```
    <Script Lauguage="JavaScript">
    <!--
      function linkGetter() {
        msgWindow = open('','','width=250,height=200')
        msgWindow.document.write("共有" + document.links.length + "个常用网站")
        for(i=0; i < document.links.length; i++) {
          msgWindow.document.write("<LI>"+document.links[i])
        }
      }
    //-->
    </Script>
  </HEAD>
  <BODY>
    常用网站：<BR>
    <A HREF="http://www.sohu.com/">搜狐</A>
    <A HREF="http://www.baidu.com/">百度</A>
    <A HREF="http://www.sina.com/">新浪</A>
    <A HREF="http://www.163.com/">网易</A>
    <INPUT TYPE="button" VALUE="网址一览" onClick="linkGetter()">
  </BODY>
</HTML>
```

本实例利用 document.link 的有关属性，实现了 document 对象上所有链接元素的遍历，再利用前面介绍过的 open()函数打开一个新的窗口来显示所获取的内容。运行并单击“网址一览”按钮之后的显示如图 6-10 所示。



图 6-10　链接对象 Link 的使用

6.4.9　表单对象 Form

表单中可以包含多种界面元素，能完成和用户交互的功能，主要包括文本对象等。以下对这些对象进行简要的说明。

1. 文本对象 Text

文本对象 Text 的属性如表 6-24 所示。

表 6-24　文本对象 Text 的属性

属 性 名 称	说　　明
defaultValue	该对象的 value 属性
form	该对象所在的表单
name	该对象的名称属性
type	该对象的类型属性
value	该对象的值属性

文本对象的方法主要有 blur()、focus()、handleEvent(事件)和 select()，其中 select()表示将该对象设置为选取状态。

文本对象支持的事件处理程序包括 onBlur、onChange、onClick、onDbClick、onFocus、onKeyDown、onKeyPress、onKeyUp、onMouseDown、onMouseUp、onMouseOver、onMouseOut、onMouseMove 和 onSelect。

表单对象 Form 是文件对象 document 的子对象。JavaScript 引擎会自动为每一个表单建立一个表单对象。其中 Forms 可以理解为一个数组，而 form 为其中的元素，每个 Form 就是一个单独的表单。通常可以采取下面的几种方式来使用表单对象：

- document.forms[索引].属性
- document.forms[索引].方法(参数)
- document.表单名称.属性
- document.表单名称.方法(参数)

表单对象 Form 的属性如表 6-25 所示。

表 6-25　表单对象 Form 的属性

属 性 名 称	说　　明
action	表单动作
elements	以索引表示的所有表单元素
encoding	MIME 的类型
length	表单元素的个数
method	方法
name	表单名称
target	目标

表单对象的方法主要有 handleEvent(事件)、reset()及 submit()；它们分别完成使事件处理程序生效、重置和提交的功能。

【实例 6-25】文本对象 Text 的用法

程序代码如 ex6_25.html 所示。

ex6_25.html

```
<HTML>
```

```
    <HEAD>
        <Script Lauguage="JavaScript">
        <!--
            function getFocus(obj) {
                obj.style.color='red';
                obj.style.background='#DBDBDB';
            }
            function getBlur(obj) {
                obj.style.color='black';
                obj.style.background='white';
            }
function isInt(elm) {
    if (isNaN(elm)) {
        alert("你输入的是" + elm + "\n 不是数字！ ");
        document.form1.pw.value = "";
        return false;
    }
    if (elm.length != 4) {
        alert("请输入四位数数字！ ");
        document.form1.pw.value = "";
        return false;
    }
}
        //-->
        </Script>
    </HEAD>
    <BODY onLoad=document.form1.name.focus()   >
        <FORM NAME="form1" onSubmit="return isInt(this.pw.value)">
            姓名：<INPUT TYPE="text" NAME="name" onFocus=getFocus(this) onBlur=getBlur(this)><BR>
            电话：<INPUT TYPE="text" NAME="tel" onFocus=getFocus(this) onBlur=getBlur(this)><BR>
            请输入四位数数字密码：<BR>
            <input type="password" name="pw" onBlur=isInt(this.pw.value)>
            <input type="submit" value="检查">
        </FORM>
    </BODY>
</HTML>
```

本实例利用 Text 对象的 onFocus 和 onBlur 事件捕捉文本框获得焦点和失去焦点的事件，再利用文本框颜色的改变，以达到提醒用户的目的；此外，对于密码框，设置了一个检查程序，这样可以对用户输入的信息进行合法性的检验。请读者自行验证运行的效果。

注意：

本实例中使用的密码对象，其使用方法基本上和文本对象相同，只是在其中录入字符时显示的为星号或圆点，以达到保密的目的，上面的实例中包含这种用法。

2. 按钮对象、提交按钮对象和重置按钮对象

这些按钮是在网页中频繁使用的，它们的外形和用法基本相同，只是由于特殊的设置，提交按钮对象和重置按钮对象具有了特殊的功能。表 6-26 中列出了按钮对象的常用属性。

表 6-26　按钮对象 Button 的属性

属 性 名 称	说　　明
form	该对象所在的表单
name	该对象的名称属性
type	该对象的类型属性
value	该对象的值属性

按钮对象的方法主要有 blur()、click()、focus()和 handleEvent(事件)。

按钮对象支持的事件处理程序包括 onBlur、onClick、onDbClick、onFocus、onKeyDown、onKeyPress、onKeyUp、onMouseDown、onMouseUp、onMouseOver、onMouseOut 和 onMouseMovet。

在 Dreamweaver 中可以方便地在网页中添加和设置按钮，在 JavaScript 中只需要按照对象的一般使用来运用就可以了。

3. 单选按钮对象 Radio

单选按钮 Radio 的主要属性如表 6-27 所示。

表 6-27　单选按钮 Radio 的属性

属 性 名 称	说　　明
checked	设置该对象为选定状态
defaultChecked	该对象的默认选定状态
form	该对象所在的表单
name	该对象的名称属性
type	该对象的类型属性
value	该对象的值属性

单选按钮对象的方法主要有 blur()、click()、focus()和 handleEvent(事件)。

单选按钮对象支持的事件处理程序包括 onBlur、onClick、onDbClick、onFocus、onKeyDown、onKeyPress、onKeyUp、onMouseDown、onMouseUp、onMouseOver、onMouseOut 和 onMouseMovet。

【实例 6-26】单选按钮对象 Radio 的用法

程序代码如 ex6_26.html 所示。

ex6_26.html

```
<HTML>
  <HEAD>
    <Script Lauguage="JavaScript">
    <!--
      function show() {
```

```
            var x = "先生";
            if (document.form1.sex[1].checked)
                x = "小姐";
            alert(document.form1.name.value + x);
        }
    //-->
    </Script>
  </HEAD>
  <BODY>
    <FORM NAME=form1>
      姓名：<INPUT TYPE="text" NAME="name"><BR>
      你是：<INPUT TYPE="radio" NAME="sex" CHECKED>帅哥
      <INPUT TYPE="radio" NAME="sex">美女<BR>
      <INPUT TYPE="button" VALUE="请单击" onClick=show()>
    </FORM>
  </BODY>
</HTML>
```

本实例利用 Radio 对象让用户选择性别，当用户单击“请单击”按钮时调用 show()函数对 Radio 对象 sex 的 checked 属性进行判断，根据结果显示出正确的性别。在浏览器中的运行结果如图 6-11 所示。

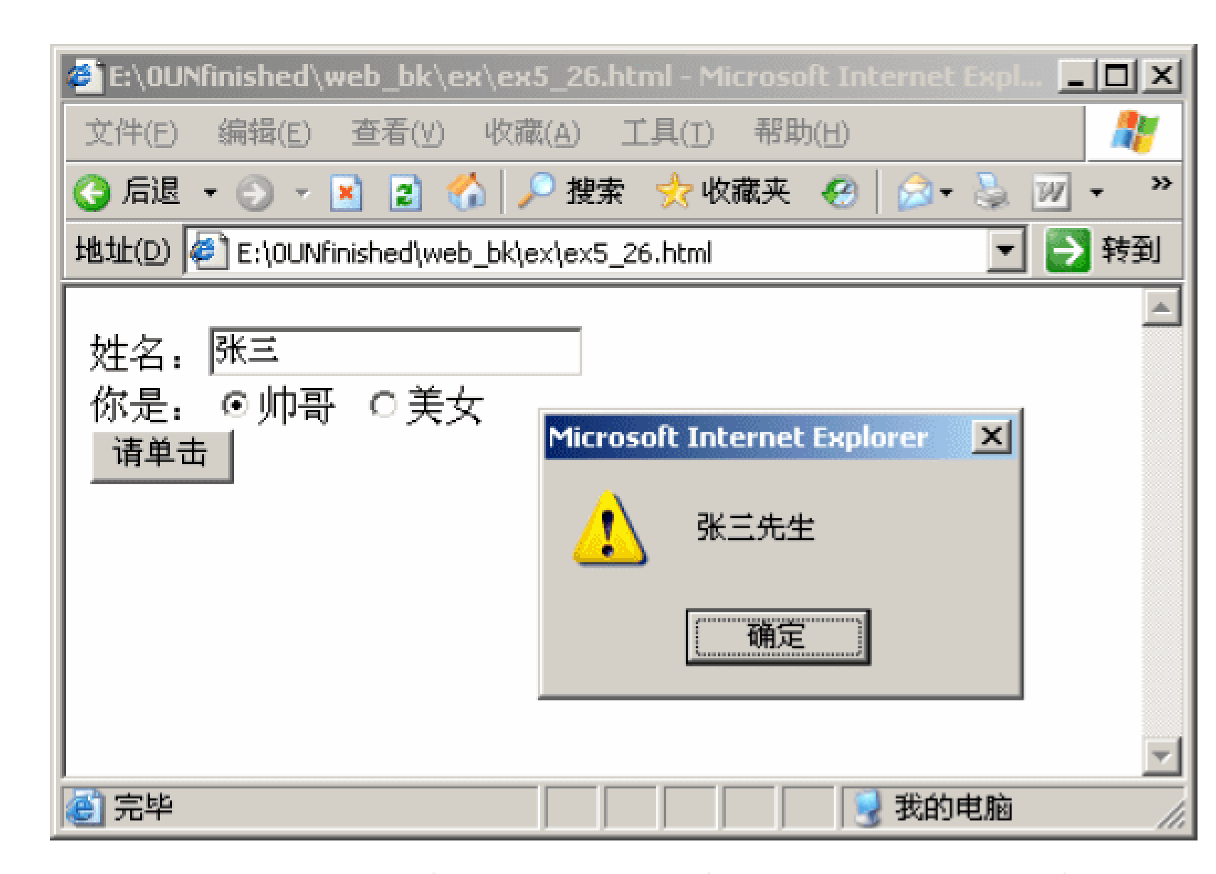


图 6-11　单选按钮对象 Radio 的用法

4. 复选框对象 Checkbox

复选框对象 Checkbox 的主要属性如表 6-28 所示。

表 6-28　复选框对象 Checkbox 的属性

属 性 名 称	说　　明
checked	设置该对象为选定状态
defaultChecked	该对象的默认选定状态
form	该对象所在的表单
name	该对象的名称属性
type	该对象的类型属性
value	该对象的值属性

复选框对象的方法主要有 blur()、click()、focus()和 handleEvent(事件)。

复选框对象支持的事件处理程序包括 onBlur、onClick、onDbClick、onFocus、onKeyDown、onKeyPress、onKeyUp、onMouseDown、onMouseUp、onMouseOver、onMouseOut 和 onMouseMovet。

注意：

其实大部分控件类对象的属性和所支持的方法是类似的，它们的区别主要在于不同控

件间特殊的那一部分，在学习时可以着重留意这些不同点。

【实例 6-27】复选框对象 Checkbox 的用法

程序代码如 ex6_27.html 所示。

ex6_27.html

```
<HTML>
  <HEAD>
    <Script Lauguage="JavaScript">
    <!--
      function count() {
        var checkCount=0;
        var num = document.form1.elements.length;
        for (var i=0; i<num; i++) {
          if (document.form1.elements[i].checked)
          checkCount++;
        }
        alert ("你喜欢  "+ checkCount + "种颜色。")
      }
    //-->
    </Script>
  </HEAD>
  <BODY>
    <FORM NAME=form1>
      请选择你喜欢的颜色：<BR>
      <INPUT TYPE="checkbox" NAME="red">红色
      <INPUT TYPE="checkbox" NAME="green">绿色
      <INPUT TYPE="checkbox" NAME="blue">蓝色<BR>
      <INPUT TYPE="button" VALUE="请单击" onClick=count()>
    </FORM>
  </BODY>
</HTML>
```

本实例利用复选框对象 Checkbox 让用户选择不同的颜色，当用户单击“请单击”按钮时调用 count()函数对 Checkbox 中属性 checked 为真的对象个数进行统计，最后将统计的结果显示在浏览器中，运行结果如图 6-12 所示。

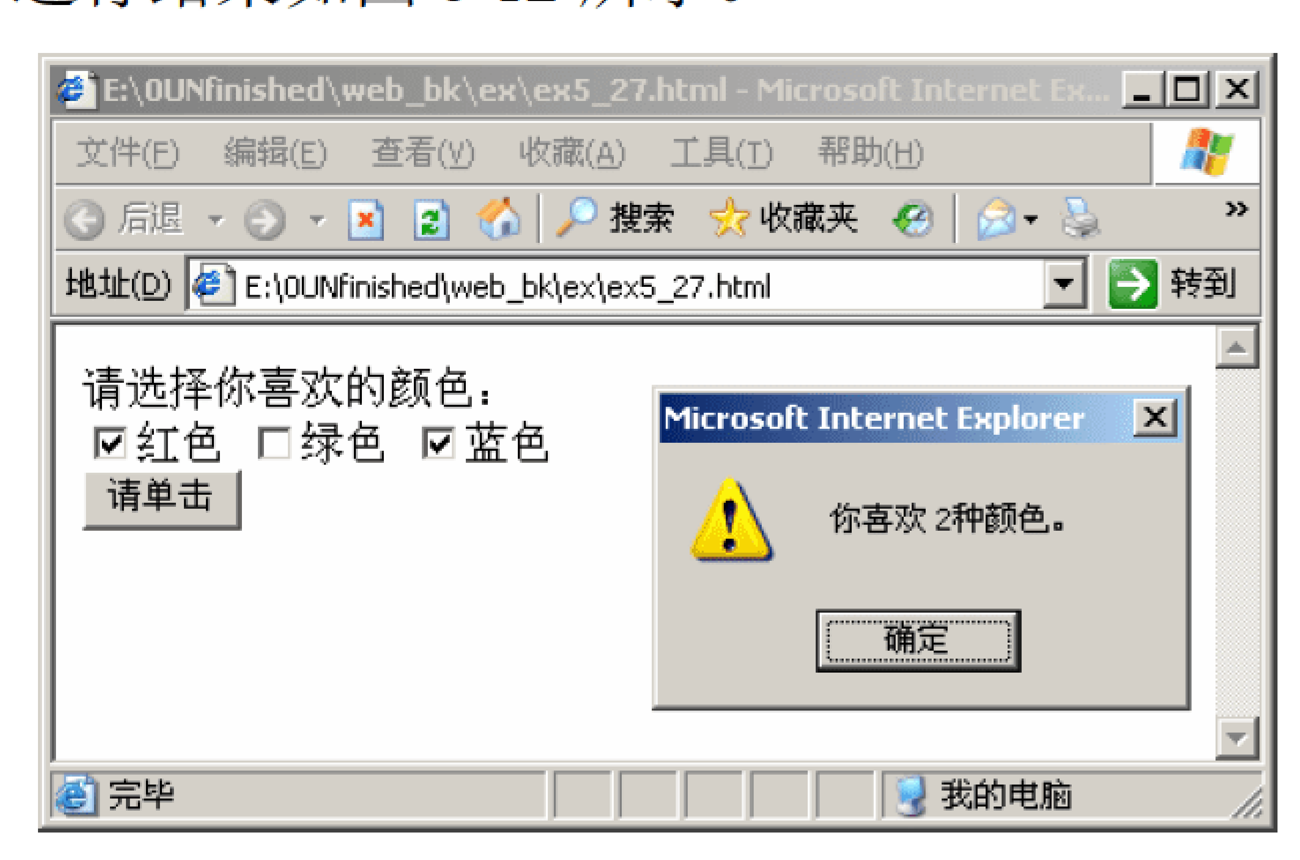


图 6-12　复选框对象 Checkbox 的用法

5. 选择对象 Select 和选项对象 Option

选择对象 Select 通常和选项对象 Option 连用，其中 Option 是 Select 的子对象。选择对象 Select 常用的属性如表 6-29 所示，而选项对象 Option 常用的属性如表 6-30 所示。

表 6-29　选择对象 Select 的属性

属 性 名 称	说　明
form	该对象所在的表单
name	该对象的名称属性
length	选项的数目
option	<option>标签
selectedIndex	所选项目的索引值
type	该对象的类型属性

表 6-30　选项对象 Option 的属性

属 性 名 称	说　明
selected	判断该选项是否被选取
defaultSelected	指定该选项为默认选定状态
index	所有选项所构成的数组索引值
length	选项的数目
text	该选项显示的文字
value	所选选项传到服务器的值

选择对象 Select 的方法主要有 blur()、focus()和 handleEvent(事件)。

选择对象 Select 支持的事件处理程序包括 onBlur、onClick、onChange、onFocus、onKeyDown、onKeyPress、onKeyUp、onMouseDown、onMouseUp、onMouseOver、onMouseOut 和 onMouseMovet。

通常在 HTML 中它们的用法如下：

```
<option value="值" selected>文字</option>
new Option([文字[,值[,defaultSelected[,selected]]]])
```

【实例 6-28】选择对象 Select 和选项对象 Option 示例

程序代码如 ex6_28.html 所示。

ex6_28.html

```
<HTML>
  <HEAD>
    <Script Lauguage="JavaScript">
    <!--
      function createOptions(){
        sel1 = document.form1.select1;
        sel2 = document.form1.select2;
```

```
            var num = sel1.selectedIndex;
            if (num > 1) {
                var option = new Option(sel1.options[num].text);
                var item = sel2.options.length;
                sel2.options[item] = option;
            }
            sel1.selectedIndex = 10000;
        }
        function delOptions() {
            var num = document.form1.select2.selectedIndex;
            if (num>1)
                document.form1.select2.options[num] = null;
            else
                document.form1.select2.selectedIndex = 10000;
        }
    //-->
    </Script>
  </HEAD>
  <BODY>
    <FORM name="form1">
        <select name="select1" size="10" onDblClick="createOptions()">
            <option>可选择项目  <option>
            <option value="幼儿园">幼儿园
            <option value="小学">小学
            <option value="中学">中学
            <option value="大学">大学
        </select>
        <input type="button" value="选择" onClick="createOptions()">
        <select name="select2" size="10">
            <option>选择项目  <option>
        </select>
        <input type="button" value="删除" onClick="delOptions()">
    </FORM>
  </BODY>
</HTML>
```

本实例利用选择对象 Select 和选项对象 Option 让用户在不同的选项中进行多项选择，用户不仅可以将已选中的项目在右侧显示，还可以在右边删除。此处分别使用了 createOptions() 和 delOptions()两个函数来实现，此外还使用了动态建立 Option 对象的功能，最终在浏览器中的运行结果如图 6-13 所示。

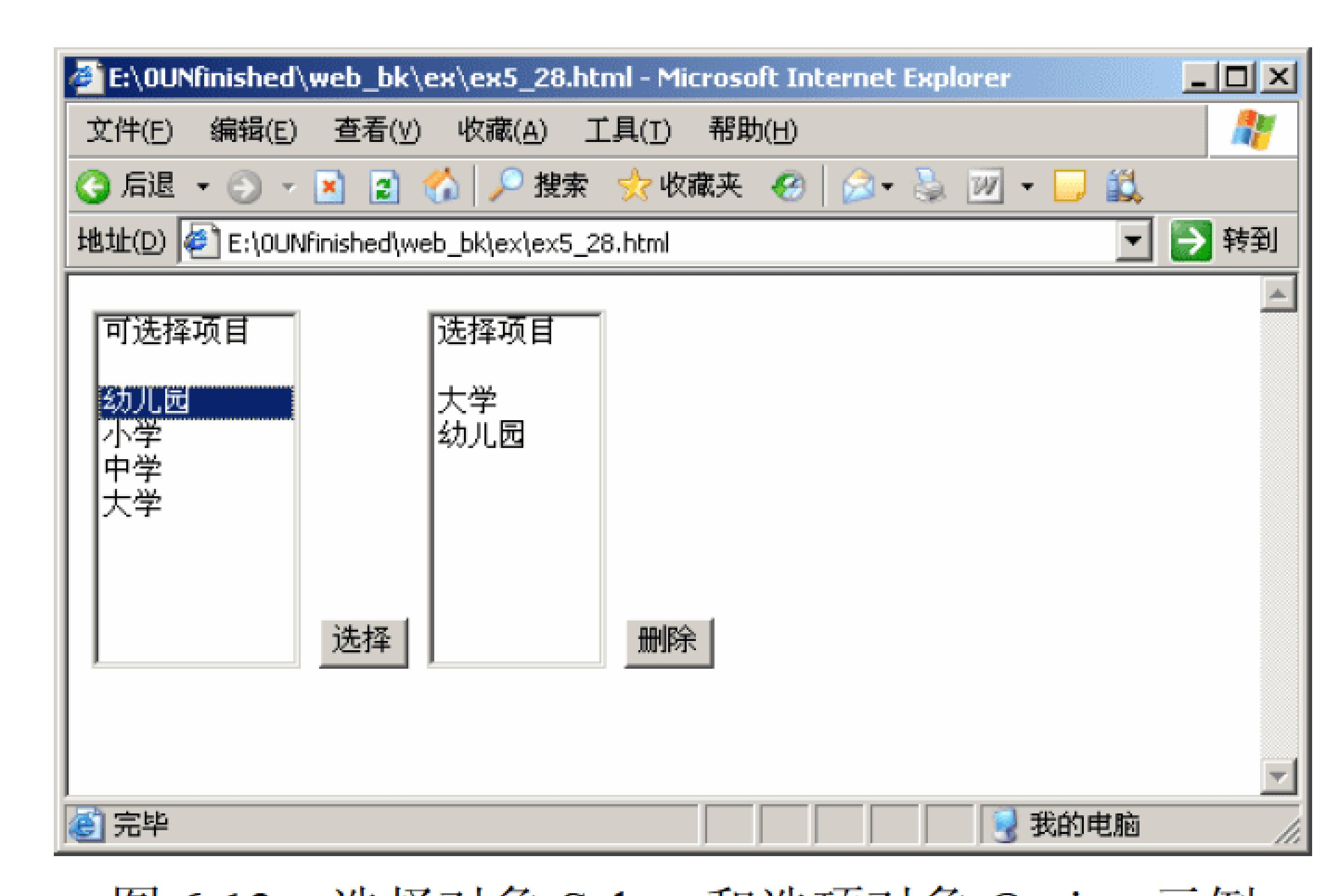


图 6-13　选择对象 Select 和选项对象 Option 示例

6. 文本区域对象 Textarea

文本区域对象 Textarea 的主要属性如表 6-31 所示。

表 6-31　文本区域对象 Textarea 的属性

属 性 名 称	说　明
defaultValue	该对象的默认值
form	该对象所在的表单
name	该对象的名称属性
type	该对象的类型属性
value	该对象的值属性

文本区域对象的方法主要有 blur()、click()、focus()和 handleEvent(事件)。

文本区域对象支持的事件处理程序包括 onBlur、onClick、onChange、onSelect、onFocus、onKeyDown、onKeyPress、onKeyUp、onMouseDown、onMouseUp、onMouseOver、onMouseOut 和 onMouseMovet。

【实例 6-29】文本区域对象 Textarea 的用法

程序代码如 ex6_29.html 所示。

ex6_29.html

```
<HTML>
  <HEAD>
    <Script Lauguage="JavaScript">
    <!--
      function isTooLong(elm){
        if (elm.length > 50) {
          alert("留言内容太长，请修改后再发送....");
          return false;
        }
      }
    //-->
    </Script>
  </HEAD>
  <BODY>
    <FORM onSubmit="return isTooLong(this.msg.value)">
      <TEXTAREA NAME="msg" COLS="30" ROWS="5" onFocus="this.value=""">欢迎留言，不过请
长话短说....
      </textarea><BR>
      <INPUT TYPE="submit" VALUE="留言完毕">
    </FORM>
  </BODY>
</HTML>
```

本实例利用文本区域对象 Textarea 的属性 length 来判断其中文字的长度，如果超过了 50 个字，则认为用户留言过多，拒绝用户提交。请读者运行本实例后自行验证。

7. 文件上传对象 FileUpload

文件上传对象 FileUpload 的主要属性如表 6-32 所示。

表 6-32　文件上传对象 FileUpload 的属性

属 性 名 称	说　　明
form	该对象所在的表单
name	该对象的名称属性
type	该对象的类型属性
value	该对象的值属性

文件上传对象的方法主要有 blur()、click()、focus()和 handleEvent(事件)。

文件上传对象支持的事件处理程序包括 onBlur、onClick、onSelect、onFocus、onKeyDown、onKeyPress、onKeyUp、onMouseDown、onMouseUp、onMouseOver、onMouseOut 和 onMouseMovet。

【实例 6-30】文件上传对象 FileUpload 示例

程序代码如 ex6_30.html 所示。

ex6_30.html

```
<HTML>
  <BODY>
    <FORM NAME="form1">
      请选择文件: <INPUT TYPE="file" NAME="oneUploadObject">
      <P>获取属性<BR>
      <INPUT TYPE="button" VALUE="获取名称" onClick="alert('名称为: ' +
document.form1.oneUploadObject.name)">
      <INPUT TYPE="button" VALUE=" 获取值 " onClick="alert('值为: ' +
document.form1.oneUploadObject.value)"><BR>
    </FORM>
  </BODY>
</HTML>
```

本实例利用文件上传对象 FileUpload 完成文件的上传，通过 name 属性和 value 属性来取得对象的名称和值。请读者运行后自行验证。

6.4.10　Cookie 对象

Cookie 对象是一种用户数据信息(Cookie 数据)，它以文件(Cookie 文件)的形式保存在客户端的 Cookies 文件夹中。Cookie 文件由所访问的 Web 站点建立，可以长久保存客户端与 Web 站点间的会话数据，且该 Cookie 数据只允许由被访问的 Web 站点来读取。

Cookie 文件内部的格式通常如下：

```
document.cookie = " 关键字 = 值 [ ; expires = 有效日期 ] [;...]"
```

有效的日期格式为 Wdy,DD-Mon-YY HH:MM:SS GMT。其中的 Wdy / Mon 表示英文星期/月份；还可以包含路径、域和安全属性；每个 Web 站点(domain)可建立 20 个 Cookie 数据；每个浏览器可存储 300 个 Cookie 数据，4K 字节；而客户可以通过浏览器等的安全

设置禁止 Cookie 数据的写入。

【实例 6-31】Cookie 对象使用示例

程序代码如 ex6_31.html 所示。

ex6_31.html

```
<HTML>
  <HEAD>
    <Script Lauguage="JavaScript">
    <!--
      var today = new Date();
      var expireDay = new Date();
      var msPerMonth = 24*60*60*1000*31;
      expireDay.setTime( today.getTime() + msPerMonth );
      document.cookie = "name=wwm;expires=" + expireDay.toGMTString();
      document.write("已经将 Cookie 写入你的硬盘中了!<br>");
      document.write("内容是：", document.cookie, "<br>");
      document.write("这个 Cookie 的有效时间是：");
      document.write(expireDay.toGMTString());
    //-->
    </Script>
  </HEAD>
</HTML>
```

本实例利用 Cookie 对象，执行写入 Cookie 文件的操作，并通过 document.cookie 将刚刚写入的值显示出来，运行后的界面显示如图 6-14 所示。

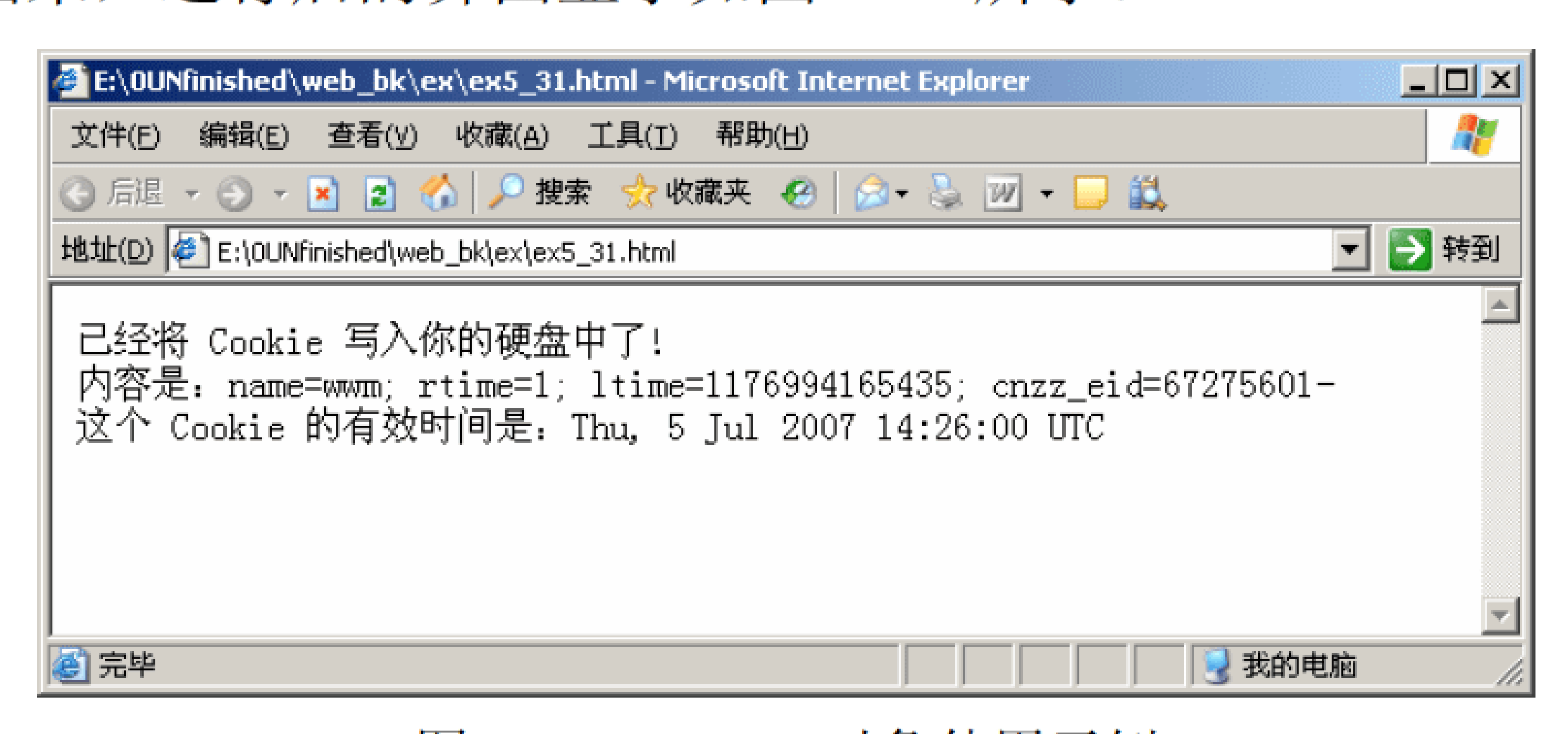


图 6-14　Cookie 对象使用示例

6.5　JavaScript 实例

6.5.1　文字连续闪烁效果

【实例 6-32】文字连续闪烁效果

本实例希望生成一个文字不断闪烁变化的网页效果，能起到醒目的作用，程序代码如 ex6_32.html 所示。

ex6_32.html

```
<HTML>
   <SCRIPT language="JAVASCRIPT">
      colors2 = new Array(6);
      colors2[0]="#000000";
      colors2[1]="#333300";
      colors2[2]="#665500";
      colors2[3]="#997700";
      colors2[4]="#CC9900";
      colors2[5]="#FFCC00";

      var i=0;
      function fLi2() {
         line2.style.visibility = "visible";
         if (i<6) {
            line2.style.color = colors2[i];
            i++;
            timerID2 = setTimeout( "fLi2()", 100);
         }
         else {
            i=0;
            line2.style.visibility="hidden";
            TimerID2=setTimeout("fLi2()",1500);
         }
      }
   </SCRIPT>

   <BODY bgcolor="#2e84ff" onload="TimerID2=setTimeout('fLi2()',500)"
onunload="clearTimeout(TimerID2)">
      <DIV id=line2 style="font-size=35px; color=black; visibility=hidden;">
      JavaScript 闪烁字
      </DIV>
   </BODY>
<HTML>
```

本实例利用自定义函数 fLi2()实现了字体颜色的变化，用 setTimeout()控制时间，在 colors2 数组中保存了 6 种颜色，利用循环使它们连续出现，由此形成闪烁字的效果。请读者自行验证。

6.5.2　旋转变幻文字效果

【实例 6-33】旋转变幻文字效果

这个实例希望生成一个能将生成字体变形的效果，程序代码如 ex6_33.html 所示。

ex6_33.html

```
<HTML>
   <HEAD>
      <Script Lauguage="JavaScript">
```

```
    <!--
      Phrase="Web 开发基础知识"
      Balises=""
      Taille=40;
      Midx=100;
      Decal=0.5;
      Nb=Phrase.length;
      y=-10000;
      for (x=0;x<Nb;x++){
        Balises=Balises + '<DIV Id=L' + x + ' STYLE="width:3;font-family: Courier
New;font-weight:bold;position:absolute;top:40;left:50;z-index:0">' + Phrase.charAt(x) + '</DIV>' }
      document.write (Balises);
      Time=window.setInterval("Alors()",10);
      Alpha=5;
      I_Alpha=0.05;
      function Alors(){
        Alpha=Alpha-I_Alpha;
        for (x=0;x<Nb;x++){
          Alpha1=Alpha+Decal*x;
          Cosine=Math.cos(Alpha1);
          Ob=document.all("L"+x);
          Ob.style.posLeft=Midx+100*Math.sin(Alpha1)+50;
          Ob.style.zIndex=20*Cosine;
          Ob.style.fontSize=Taille+25*Cosine;
          Ob.style.color="rgb("+ (27+Cosine*80+50) + ","+ (127+Cosine*80+50) + ",0)";
        }
      }
    //-->
    </Script>
  </HEAD>
</HTML>
```

本实例生成旋转变幻文字，它利用 Alors()函数实现了文字的变化，其中的循环生成了连续变化的数值，在函数内部使用算术对象中的一些三角函数实现了文字的大小、位置和颜色的计算，最终实现了旋转变幻文字的效果。实际运行的效果如图 6-15 所示。

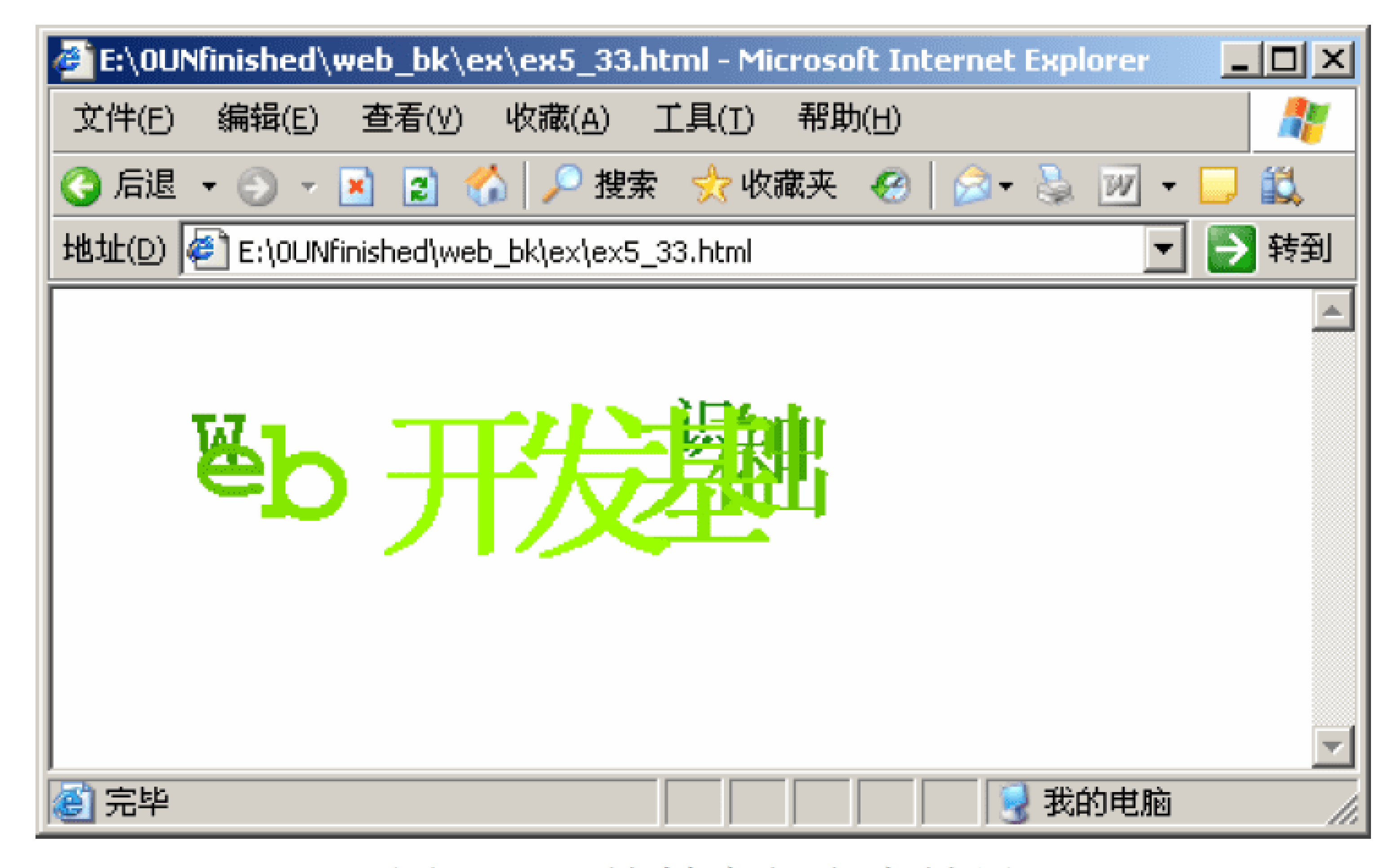


图 6-15　旋转变幻文字效果

6.5.3　图片广告轮显的实现

【实例 6-34】图片广告轮显的实现

图片轮显是很多网站中都采用的一种利用有限显示区域来展示多幅图片的技术手段，它可以实现图片广告的自动切换以及图片幻灯效果，程序代码如 ex6_34.html 和 5ads.js 所示。

ex6_34.html

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
    <meta name="description" content="JS 图片轮显广告" />
    <title>JS 图片轮显广告</title>
    <style type="text/css">
      body{text-align:center;}
      td {font-size: 12px;}
      #textslide{color:#333333;}
    </style>
  </head>
  <BODY>
    <table width=325 border=0 cellpadding=0 cellspacing=0>
      <tr valign=top>
        <td colspan=3>
     <a onClick="gotoshow()" onMouseOver="tu_ove()" onMouseOut="ou()" style="cursor:hand">
       <img src="images/ad-01.jpg" width=325 height=190 name="slide"   border=0>
     </a>
   </td>
 </tr>
      <tr>
        <td width=229 height="19" align=center bgcolor=#f4f4f4 class="white">
     <div id=textslide>焦点图标题层</div>
   </td>
   <td width=1 bgcolor=#7C7C7C>
          <div style="position:relative">
            <div style="position:absolute;top:10px">
              <table width=95 border=0 cellpadding=0 cellspacing=0>
                <tr valign=top align=center>
                  <td width="19" height="0">
                    <div style="position:relative">
                      <div id=xiaotu1 style="position:absolute;top:-19px;left:0px">
                    <img src=images/bian1.gif id=tu1 width=10 height=3 border=0>
                  </div>
                </div>
                  </td>
                  <td width="19" height="0">
                    <div style="position:relative">
                      <div id=xiaotu2 style="position:absolute;top:-19px;left:0px">
                    <img src=images/bian1.gif id=tu2 width=10 height=3>
                  </div>
```

```
                    </div>
                        </td>
                        <td width="19" height="0">
                            <div style="position:relative">
                                <div id=xiaotu3 style="position:absolute;top:-19px;left:0px">
                            <img    src=images/bian1.gif id=tu3 width=10 height=3>
                        </div>
                    </div>
                        </td>
                        <td width="19" height="0">
                            <div style="position:relative">
                                <div id=xiaotu4 style="position:absolute;top:-19px;left:0px">
                            <img src=images/bian1.gif id=tu4 width=10 height=3>
                        </div>
                    </div>
                        </td>
                        <td width="19" height="0">
                            <div style="position:relative; left: 1px;">
                                <div id=xiaotu5 style="position:absolute;top:-19px;left:0px">
                            <img src=images/bian1.gif id=tu5 width=10 height=3>
                        </div>
                    </div>
                        </td>
                    </tr>
                </table>
            </div>
        </div>
    </td>
    <td width=95 height="19">
        <table width=95 border=0 cellpadding=0 cellspacing=0>
            <tr valign=top>
                <td width="19" height="19" class="homejdboder">
        <a style="cursor:hand" onMouseOver="ove(0)" onMouseOut="ou()">
            <img src="images/1.gif" width="19" height="19" border=0>
        </a>
    </td>
    <td width="19" height="19" class="homejdboder">
        <a style="cursor:hand" onMouseOver="ove(1)" onMouseOut="ou()">
            <img src="images/2.gif" width="19" height="19" border=0>
        </a>
    </td>
    <td width="19" height="19" class="homejdboder">
        <a style="cursor:hand" onMouseOver="ove(2)" onMouseOut="ou()">
            <img src="images/3.gif" width="19" height="19" border=0>
        </a>
    </td>
    <td width="19" height="19" class="homejdboder">
        <a style="cursor:hand" onMouseOver="ove(3)" onMouseOut="ou()">
            <img src="images/4.gif" width="19" height="19" border=0>
        </a>
    </td>
    <td width="19" height="19">
```

```
                <a style="cursor:hand" onMouseOver="ove(4)" onMouseOut="ou()">
                    <img src="images/5.gif" width="19" height="19" border=0>
                </a>
            </td>
        </tr>
            </table>
        </td>
    </tr>
</table>
<script language=JavaScript src=" 5ads.js"></script>
</body>
</html>
```

5ads.js

```
// slideimages 数组为变幻的图
var slideimages=new Array();
slideimages[0]="images/ad-01.jpg";
slideimages[1]="images/ad-02.jpg";
slideimages[2]="images/ad-03.jpg";
slideimages[3]="images/ad-04.jpg";
slideimages[4]="images/ad-05.jpg";
// slidetext 数组中保存图片下方需要显示的文字
var slidetext=new Array();
slidetext[0]="图 01";
slidetext[1]="图 02";
slidetext[2]="图 03";
slidetext[3]="图 04";
slidetext[4]="图 05";
// slidetext 数组为单击每个大图后跳转的 URL
var slidelinks=new Array();
slidelinks[0]="http://www.njupt.edu.cn";
slidelinks[1]="http://www.njupt.edu.cn";
slidelinks[2]="http://www.njupt.edu.cn";
slidelinks[3]="http://www.njupt.edu.cn";
slidelinks[4]="http://www.njupt.edu.cn";
//  焦点图初始内容——start
var slidespeed=3000
var slidesanjiaoimages=new Array("images/bian2.gif","images/bian1.gif");
var slidesanjiaoimagesname=new Array("tu1","tu2","tu3","tu4","tu5");
var filterArray=new Array();
filterArray[0]="progid:DXImageTransform.Microsoft.Pixelate
(enabled=false,duration=2,maxSquare=25 )";
filterArray[1]="progid:DXImageTransform.Microsoft.Stretch (duration=1,stretchStyle=PUSH)";
filterArray[2]="progid:DXImageTransform.Microsoft.Stretch(duration=1)";
filterArray[3]="progid:DXImageTransform.Microsoft.Slide(bands=8, duration=1)";
filterArray[4]="progid:DXImageTransform.Microsoft.Fade ( duration=1,overlap=0.25 )";
var imageholder=new Array()
var ie55=window.createPopup
for(i=0;i<slideimages.length;i++){
    imageholder[i]=new Image()
    imageholder[i].src=slideimages[i]
```

```
}
function tu_ove(){
    clearTimeout(setID)
}
function ou(){
    slideit()
}
var whichlink=0
var whichimage=0
function gotoshow(){
    window.open(slidelinks[whichlink]);
}
function slideit(){
    document.images.slide.style.filter=filterArray[whichimage];
    pixeldelay=(ie55)? (document.images.slide.filters[0].duration*1000) : 0
    if (!document.images) return
    if (ie55) {
        document.images.slide.filters[0].apply();
        document.images.slide.filters[0].play();
    }
    document.images.slide.src=imageholder[whichimage].src
    document.getElementById("textslide").innerText=slidetext[whichimage];
    document.getElementById("tu1").src=slidesanjiaoimages[0];
    document.getElementById("tu2").src=slidesanjiaoimages[0];
    document.getElementById("tu3").src=slidesanjiaoimages[0];
    document.getElementById("tu4").src=slidesanjiaoimages[0];
    document.getElementById("tu5").src=slidesanjiaoimages[0];
    document.getElementById(slidesanjiaoimagesname[whichimage]).src=slidesanjiaoimages[1];
    if(ie55)
        document.images.slide.filters[0].play()
    whichlink=whichimage
    whichimage=(whichimage<slideimages.length-1)? whichimage+1:0
    setID=setTimeout("slideit()",slidespeed+pixeldelay)
}
slideit()
function ove(n){
    clearTimeout(setID)
    whichimage=n;
    document.images.slide.src=imageholder[whichimage].src
    document.getElementById("textslide").innerText=slidetext[whichimage];
    document.getElementById("tu1").src=slidesanjiaoimages[0];
    document.getElementById("tu2").src=slidesanjiaoimages[0];
    document.getElementById("tu3").src=slidesanjiaoimages[0];
    document.getElementById("tu4").src=slidesanjiaoimages[0];
    document.getElementById("tu5").src=slidesanjiaoimages[0];
    document.getElementById(slidesanjiaoimagesname[whichimage]).src=slidesanjiaoimages[1];
}
```

本实例能生成一个 5 幅图片轮显的页面广告，当鼠标不在此广告区域时，图片会实现自动轮显且具有随机切换效果，当鼠标移入本区域后，会停止自动切换，当鼠标单击右下方的彩色数字区域时，可实现图片的点播。实际运行的效果如图 6-16 所示。

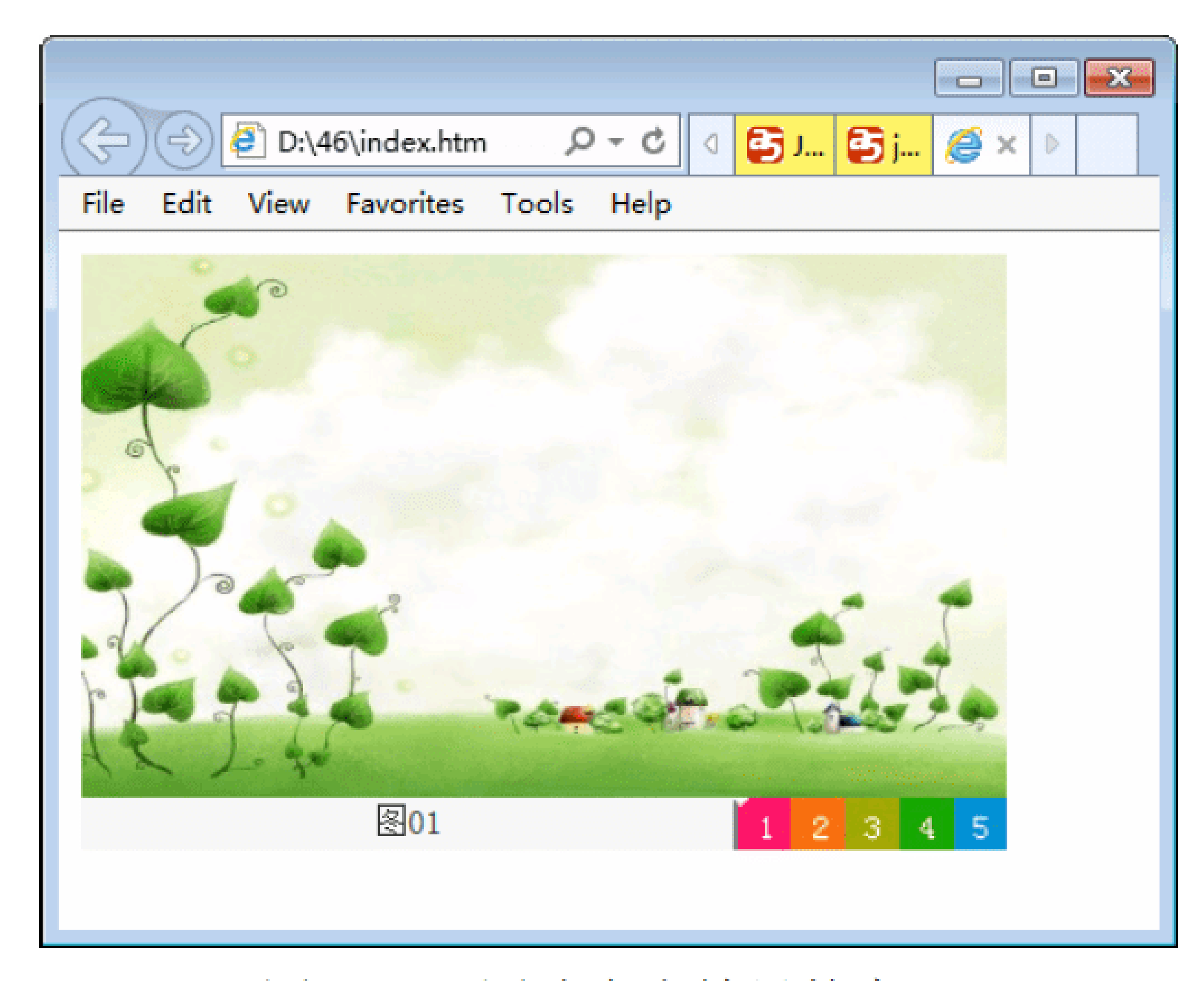


图 6-16　图片广告轮显的实现

6.5.4　一个益智小游戏的实现

【实例 6-35】一个益智小游戏的实现

游戏的制作不同于一般网页之处在于它需要具有较强的交互性及界面的联动性。本实例利用鼠标单击和界面生成技术实现了一个简单的益智游戏，其程序代码如 ex6_35.html 所示。

ex6_35.html

```
<HTML>
  <HEAD>
    <SCRIPT Lauguage="JavaScript">
    <!--
      function ShowMenu(bMenu) {
      document.all.idFinder.style.display = (bMenu) ? "none" : "block"
      document.all.idMenu.style.display = (bMenu) ? "block" : "none"
      idML.className = (bMenu) ? "cOn" : "cOff"
      idRL.className = (bMenu) ? "cOff" : "cOn"
      return false
      }
    //-->
    </SCRIPT>
    <STYLE type="text/css">
    <!--
      A.cOn {text-decoration:none;font-weight:bolder}
      #article {font: 12pt Verdana, geneva, arial, sans-serif;    background: white; color: black; padding:
10pt 15pt 0 5pt}
      #article P.start {text-indent: 0pt}
      #article P {margin-top:0pt;font-size:10pt;text-indent:12pt}
      #article #author {margin-bottom:5pt;text-indent:0pt;font-style: italic}
      #pageList P {padding-top:10pt}
      #article H3 {font-weight:bold}
      #article DL, UL, OL {font-size: 10pt}
    -->
```

```
        </STYLE>
        <SCRIPT>
        <!--
          function addList(url,desc) {
            if ((navigator.appName=="Netscape") || (parseInt(navigator.appVersion)>=4)) {
              var
w=window.open("","_IDHTML_LIST_","top=0,left=0,width=475,height=150,history=no,menubar=no,
status=no,resizable=no")
              var d=w.document
              if (!w._init) {
                d.open()
                d.write("<TITLE>Loading...</TITLE><EM>Loading...</EM>")
                d.close()
d.location.replace("/assist/listing.asp?url="+escape(url)+"&desc="+escape(desc))
                w.opener=self
                window.status="Personal Assistant (Adding): " + desc
              }
              else {
                window.status=w.addOption(url,desc)
                w.focus()
              }
            }
            else {
              alert("Your browser does not support the personal assistant.")
              return false
            }
          }
        // -->
        </SCRIPT>
        <STYLE>
          #board {cursor: default}
          #board TD {width: 25px; height: 25px; }
        .style1 {color: #FF0099}
    .style3 {color: #CC0099}
        .style4 {
    color: #330000;
    font-weight: bold;
    }
        </STYLE>
      </HEAD>
      <BODY>
        <SCRIPT>
          var size=10
          var moves = 0
          var off = size*2
          var on = 0
          var current = null
          function doOver() {
            if ((event.srcElement.tagName=="TD") && (current!=event.srcElement)) {
              if (current!=null)
                current.style.backgroundColor = current._background
              event.srcElement._background = event.srcElement.style.backgroundColor
```

```
            event.srcElement.style.backgroundColor = "lightgrey"
            current = event.srcElement
        }
    }
    function setColor(el) {
        if ((el._background=="") || (el._background==null)) {
            el.style.backgroundColor = "yellow"
            el._background = "yellow"
        } else {
            el.style.backgroundColor = ""
            el._background = ""
        }
    }
    function countLights() {
        off = 0; on = 0
        for (var x=0; x < size; x++)
            for (var y=0; y < size; y++) {
                var p = board.rows[x].cells[y]
                if (p._background=="yellow")
                    on++
                else
                    off ++
            }
        document.all.on.innerText = on
        if (off!=0)
            document.all.off.innerText = off
        else
            document.all.off.innerText = "You Win!"
        return (off==0)
    }
    function doClick() {
        setColor(current)
        var cellIdx = current.cellIndex
        var rowIdx = current.parentElement.rowIndex
        if (rowIdx>0)
            setColor(board.rows[rowIdx-1].cells[cellIdx])
        if (rowIdx<size-1)
            setColor(board.rows[rowIdx+1].cells[cellIdx])
        if (cellIdx>0)
            setColor(board.rows[rowIdx].cells[cellIdx-1])
        if (cellIdx<size-1)
            setColor(board.rows[rowIdx].cells[cellIdx+1])
        moves++
        document.all.moves.innerText = moves
        win = countLights()
        if (win) {
            board.onclick = null
            board.onmouseover = null
            current.style.background = "yellow"
        }
    }
    function buildBoard() {
```

```
        var str = "<TABLE ID=board ONSELECTSTART=\"return false\" ONCLICK=\"doClick()\"
ONMOUSEOVER=\"doOver()\" cellspacing=1 cellpadding=1 border=1>"
        for (var x=0; x < size; x++) {
            str+="<TR>"
            for (var y=0; y < size; y++) {
                str+="<TD> </TD>"
            }
            str+="</TR>"
        }
        str+="</TABLE>"
        return str
    }
    function newGame() {
        size = document.all.gameSize.value
        if (size<3)
            size=3
        if (size>15)
            size=15
        document.all.gameSize.value = size
        document.all.board.outerHTML = buildBoard()
        moves=0
        document.all.moves.innerText = moves
        countLights()
    }
</SCRIPT>
<BR>
<BR>
    <CENTER class="style1">
        <font face="隶书" size=6>小游戏</font>
    </center>
<BR>
<CENTER>
<TABLE border=5 bordercolor=blue borderlight=green>
    <tr><td align=center height=100>
    <p align="center"><big><span class="style3">在方格上单击，试试是否可以将下面的方格全部
变为黄色</span><span class="style4"><big><big>？ </big></big></span></big></p>
    <TABLE border="1" width="100%">
        <tr>
            <td width="50%"><div align="right">
                <TABLE id="score" border="0" width="284">
                    <tr>
                        <td width="52">移动次数: </td>
                        <td id="moves" width="33">0</td>
                        <td width="42">灯灭:</td>
                        <td id="off" width="36">25</td>
                        <td width="46">灯亮:</td>
                        <td id="on" width="39">0</td>
                    </tr>
                </TABLE>
            </div></td>
            <td width="50%"><div align="left">
                <TABLE width="204">
```

```
                    <tr>
                        <td width="54">大小: </td>
                        <td width="41"><input id="gameSize" type="text" value="10" size="2"></td>
                        <td width="97"><input onClick="newGame()" type="button" value="开始游戏
"></td>
                    </tr>
                </TABLE>
            </div>
        </td>
    </tr>
</TABLE>
<p align="center">
<SCRIPT>
    document.write(buildBoard())
</SCRIPT></p>
</BODY>
</HTML>
```

本实例在<BODY>中利用 buildBoard()函数的返回参数建立了一个表格，这个表格构成了游戏的主显示区域，上方的区域中显示了当前游戏的统计数据，包括移动次数、灯灭、灯亮，还可以对游戏中灯的数量进行控制。这个例子的代码复杂，它综合利用了各种技术，请读者仔细分析，实际运行的效果如图 6-17 所示。

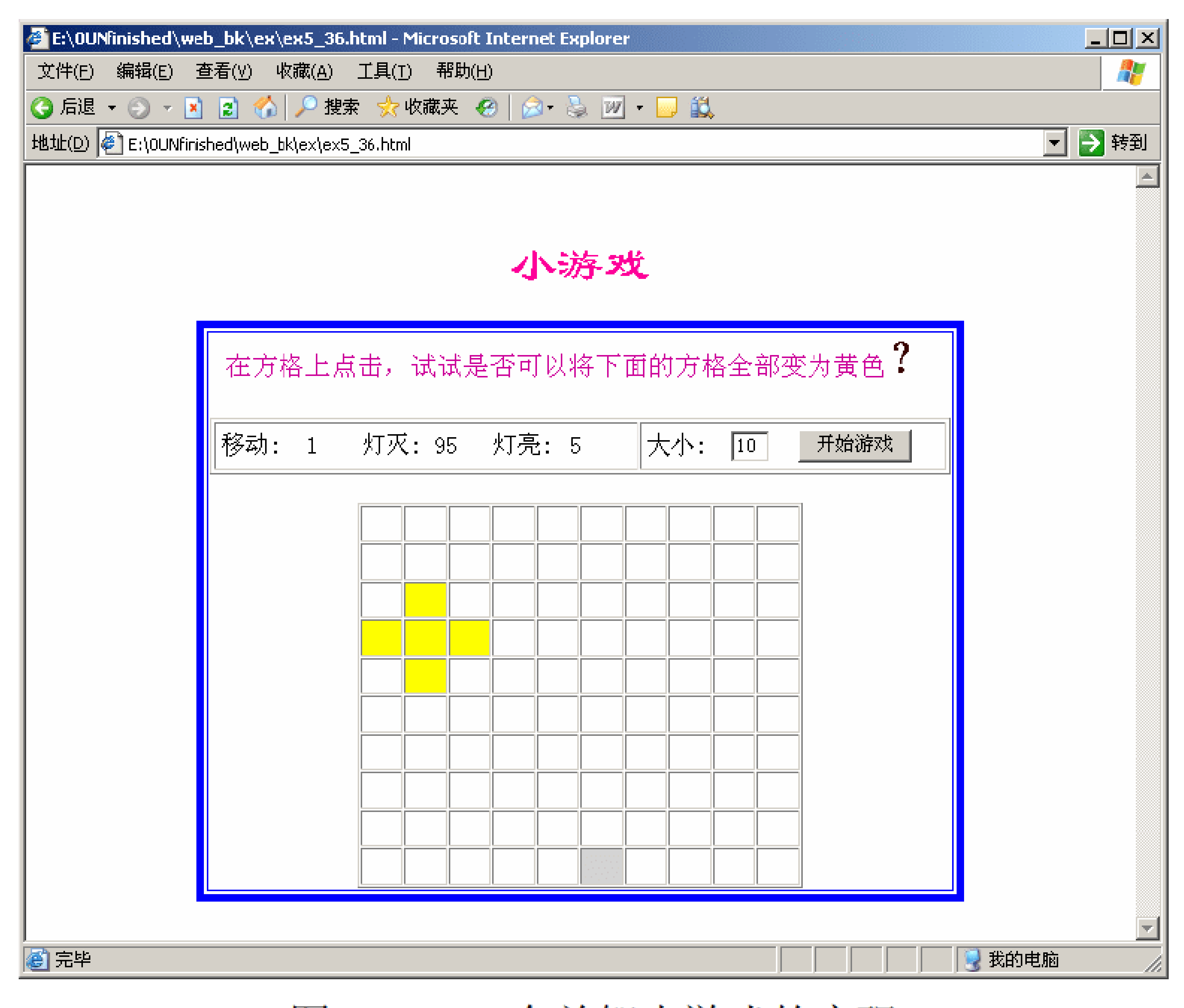


图 6-17　一个益智小游戏的实现

6.6　本章小结

本章讲解了网页制作中的一个重要部分——JavaScript 的有关知识，JavaScript 脚本语言是一种功能强大的网页编程语言，它是通过嵌入到 HTML 文件中的机制来工作的，主要用于增强网页的功能和表现力；由于在客户端运行，因此无须像服务器端技术那样等待网

络传输，运行速度更快。

通过本章的学习，可以掌握 JavaScript 语言的基本概念和基本语法，及在网页中插入脚本语言的几种方式；深刻理解有关函数中变量的作用域和各类控制语句的功能；理解和灵活运用 JavaScript 中常用的几个对象的属性和方法，包括 JavaScript 内置对象和文档对象模型。

本章最后的几个实例综合运用了上面各个部分介绍的知识，通过阅读和理解这些实例，可以深化学习，达到灵活运用的目的。

6.7　思考和练习

1. JavaScript 的运行机制是怎样的？
2. 本章所介绍的 3 种将 JavaScript 代码引入网页的方式是否存在差异？
3. 如何在 JavaScript 中使输出的 float 类型的数据保留两位小数？
4. 3 种循环结构之间的差别在哪里？
5. 在用户浏览网页时，如何实现通过单击将本网站加入收藏夹？
6. 如何在页面上利用单击某个按钮或超链接来关闭浏览窗口？

第7章　服务器端开发——动态网页技术基础

动态网页基于前面各章节中所介绍的知识，是构建完整、实用网站的基础，与 JavaScript 不同的是，本章所介绍的开发和运行环境都是基于服务器的。本章讲述构建动态网页的各种主要技术，阐明动态网页运行的基本原理，并通过介绍多种开发技术以及相应的应用范例，向读者全方位地介绍动态网页技术。本章所涉及的开发技术包括了历史上和目前应用最广、最为成熟的几种：CGI、ASP、ASP.NET、JSP、PHP、ISAPI/NSAPI、Java Servlet 和 Java Applet 等，并对它们的不同特点进行了比较，便于读者熟悉它们各自的优缺点以利于在实际项目开发时作出正确选择。通过本章的学习，读者可以对动态网页技术有一个全面概括的了解；通过对各种流行的动态网页技术之间的比较，来帮助读者选择适合的开发技术。

本章要点：

- 动态网页的基本特点
- .NET 动态网页的基本开发方法
- Java 技术基础
- 不同动态网页开发技术的异同

7.1　动态网页基本原理

这里所说的动态网页并不是指在网页上由于放入了一些诸如 Flash 等的动画，而使网页呈现内容能实时变化的网页。“动态”的“动”指的是“交互性”，通俗的说就是网页能不能根据访问者或访问时间的不同而显示出不同的内容，即本书1.3.1 节中所介绍的有关“活动页面”的内容。

单纯利用静态 HTML 开发的 Web 站点虽然开发周期短、开发难度低，且可以实现足够精美的页面，但由于难以适应信息频繁更新以及交互的需求，存在先天的不足。比如，静态网页无法根据用户在客户端浏览器中所输入的参数，在服务器对数据查询后再将符合条件的数据集回传给客户端浏览器，而动态网页技术弥补了这一不足。

动态网页可分为客户端动态网页和服务器端动态网页两类，下面简要介绍一下它们各自的工作原理。

1. 客户端动态网页

在客户端模型中，附加到浏览器上的模块(插件)完成创建动态网页的全部工作。HTML代码通常随含有一套指令的文件传送到浏览器，此文件在 HTML 页中引用。还有一种情况，是这些指令与 HTML 代码混合在一起，当遇到用户请求时，浏览器利用这些指令生成纯 HTML。也就是说，用户看到的网页是根据用户的请求动态生成然后返回到浏览器的。

客户端技术在近年来越来越不受欢迎，因为使用该技术需要下载客户端软件，而且当需要下载其他单独的指令文件时所需时间较长。另外，因为每一种浏览器都以不同的方式解释指令，所以不能保证 Internet Explorer 能正确理解指令，同时其他不同的浏览器如 Chrome、FireFox 或 Opera 也能够理解它们。

客户端技术的另一个缺点是在使用服务器资源的客户端代码时会出现安全性等方面的问题，因为如果代码是在客户端被解释执行的，那么客户端的代码将会完全公开，这也是很多有安全性要求的网站所不希望的。

2. 服务器端动态网页

在服务器端模型中，HTML 源代码与混合在其中的一套指令存储于 Web 服务器中。当用户请求该页时，这些指令在服务器上被处理，然后再返回浏览器。与客户端模型相比，只有描述最终显示页面的 HTML 代码才被传到客户端浏览器，如果设计得足够通用，则可以保证大多数浏览器能正确显示该页。能提供这种方案的服务器端动态网页技术包括：PHP、CGI、ASP、JSP 和 ASP .NET 等。下面介绍它们共同的工作原理：

(1) 当用户请求某个 PHP(CGI、ASP、JSP 或 ASPX 等)页面时，Web 服务器响应 HTTP 请求，调用 PHP(CGI、ASP、JSP 或 ASPX 等)引擎，解释(或编译)并执行被申请的文件。

(2) 若脚本中含有访问数据库的语句，则通过 ODBC(或 ADO、OLE DB、JDBC 等连接方式)与后台数据库建立连接，再由数据库访问组件执行访问数据库的操作。

(3) PHP 等脚本在服务器端解释(某些技术采取在服务器端编译的方式)并执行，根据从数据库所获取的结果集生成符合设计需要的 HTML 网页，最终回送到客户端来响应用户请求，上述所有环节均由 WWW 服务器负责。

因此，动态网页实际上就是存放在服务器端的程序，由客户端提出执行请求，在服务器端运行，运行的结果通过 HTML 的形式传回客户端。

7.2　.NET 介绍

7.2.1　ASP.NET 简介

ASP.NET 又叫 ASP+，虽然名称类似于 ASP(Active Server Pages)，但它并不仅仅是对 ASP 的简单升级，而是微软推出的一种脚本语言。ASP.NET 是微软.NET 体系结构的一部分。

ASP.NET 在兼顾 ASP 优点的基础上，参照 Java、VB 语言的优势加入了许多新的特色。

它能支持多种编程语言，如可使用脚本语言(VBscript、Jscript、Perlscript 和 Python 等)以及编译语言(Visual Basic、C#、C、Cobol、Smalltalk 和 Lisp 等)。

7.2.2 .NET 战略

随着网络经济的到来，微软希望帮助用户能够在任何时候、任何地方、利用任何工具来获得网络上的信息，并享受网络通信所带来的方便和快捷。由此设立的.NET 战略就是为了实现上述的目标。微软公开宣布，公司将着重于网络服务和网络资源共享的开发工作，并将为公众提供更加丰富、有用的网络资源与服务。

基于此战略，微软推出的平台叫作“新一代 Windows 服务”(NGWS)，并将这个平台注册为 Microsoft .NET，于 2002 年 4 月发布。在.NET 环境中，微软不仅仅是平台和产品的开发者，并且还将作为架构服务提供商、应用程序提供商，以便全方位开展基于 Internet 的服务。

Microsoft .NET 平台的基本思想是：将侧重点从连接到互联网的单一网站或设备上，转移到计算机、设备和服务群组上，使其通力合作，提供更广泛更丰富的解决方案。用户将能够控制信息的传送方式、时间和内容。计算机、设备和服务群组将能够相辅相成，从而提供丰富的服务，而不是像从前那样，由用户提供唯一的集成。企业可以提供一种方式，允许用户将他们的产品和服务无缝地嵌入自己的电子构架中。这种思路将扩展 20 世纪 80 年代首先由 PC 所赋予用户的个人权限。

Microsoft .NET 开创了互联网的新局面，基于 HTML 的信息显示将通过 XML 战略得到增强。XML 是由“万维网联盟”(W3C)定义且受到广泛支持的行业标准，HTML 标准也是由该组织发布的。XML 提供了一种从数据的演示视图分离出实际数据的方式，这是新一代互联网的关键，能方便对信息的组织、编程和编辑，可以更有效地将数据分布到不同的数字设备，并允许各站点进行合作，提供可以相互作用的 Web 服务(Web Service)。

注意：

这里提到的 XML 将在本书第 8 章中进行更为详细的介绍。

Microsoft.NET 平台包括用于创建和操作新一代服务的.NET 基础结构和工具；可以启用大量客户机的.NET 用户体验；用于建立新一代高度分布式的数以百万计的.NET 积木式组件服务；以及用于启用新一代智能互联网设备的.NET 设备软件。

.NET 环境中的好处在于：

(1) 使用统一的 Internet 标准(如 XML)将不同的系统对接。

(2) 这是 Internet 上首个大规模的高度分布式应用服务架构。

(3) 使用了一个名为“联盟”的管理程序，这个程序能全面管理平台中运行的服务程序，并且为它们提供强大的安全保护后台。

.NET 平台包括如下组件：

(1) 用户数据访问技术，其中包括一个新的基于 XML 的、以浏览器为组件的混合信息架构，叫作“通用画板”。

(2) 基于 Windows DNA 的构建和开发工具。

(3) 一系列模块化的服务，其中包括认证、信息传递、存储、搜索和软件送递功能。

(4) 一系列驱动客户设备的软件。

7.3　ASP.NET 应用的开发实例

【实例 7-1】使用 ASP <% %> 呈现块

程序代码如 ex7_1.aspx 所示。

ex7_1.aspx

```
<html>
    <head>
        <title>ex7_1.aspx</title>
    </head>
    <body>
        <center>
        <form action="" method="post">
            <h3> 姓名：<input id="Name" type=text>
            类别：<select id="类别" size=1>
                        <option>psychology</option>
                        <option>business</option>
                        <option>popular_comp</option>
                    </select>
            </h3>
            <input type=submit value="查找">
            <p>
            <% for (int i=0; i <8; i++) { %>
                <font size="<%=i%>"> 欢迎使用 ASP.NET </font> <br>
            <% }%>
        </form>
        </center>
    </body>
</html>
```

ASP.NET 提供与现有 ASP 页语法的兼容。这包括支持可在.aspx 文件内与 HTML 内容混合的<% %>代码呈现块，这些代码块在网页呈现时按由上而下的方式执行。

上面的示例说明如何使用<% %>呈现块在 HTML 块上循环(每次增加字体大小)，该实例运行后浏览器中的显示如图 7-1 所示。

注意：

在使用 Visual Studio 进行开发时，可以将 C#程序代码写在.aspx 文件中，与 HTML 混合在一起，也可以将程序代码写在单独的同名.cs 文件中。这两种方法都可以，但 Web 开发标准中建议将实现业务逻辑的 C#代码与 HTML 代码分开放在不同的文件中，这样可以提高程序的可阅读性，将视图与业务分开。

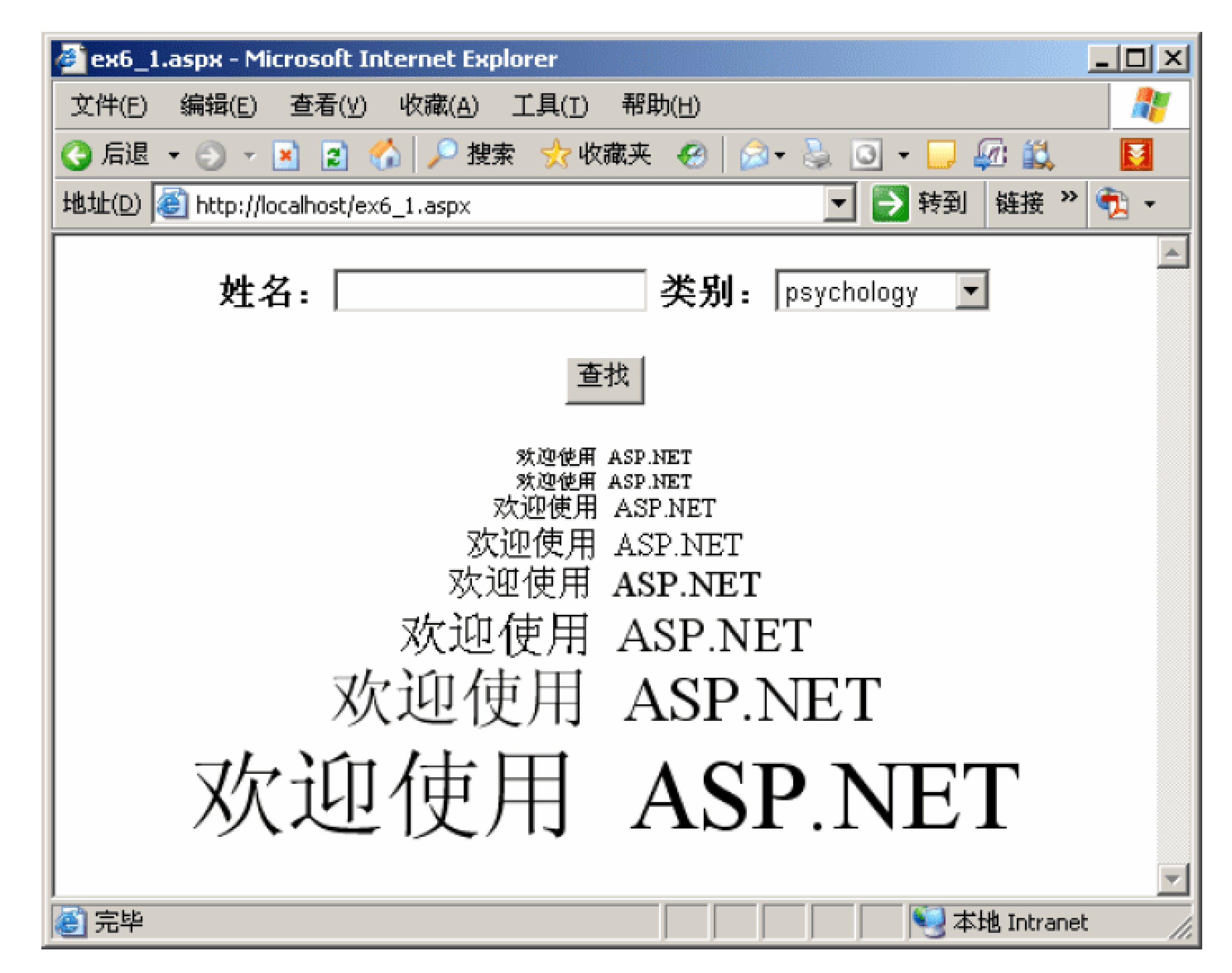


图 7-1　使用 ASP <% %> 呈现块

【实例 7-2】处理多个控件操作事件

程序代码如 ex7_2.aspx 所示。

ex7_2.aspx

```
<%@ Page Language="C#" %>
<html>
    <script language="C#" runat="server">
        void AddBtn_Click(Object Src, EventArgs E) {
            if (AvailableFonts.SelectedIndex != -1) {
                InstalledFonts.Items.Add(new ListItem(AvailableFonts.SelectedItem.Value));
                AvailableFonts.Items.Remove(AvailableFonts.SelectedItem.Value);
            }
        }
        void AddAllBtn_Click(Object Src, EventArgs E) {
            while (AvailableFonts.Items.Count != 0) {
                InstalledFonts.Items.Add(new ListItem(AvailableFonts.Items[0].Value));
                AvailableFonts.Items.Remove(AvailableFonts.Items[0].Value);
            }
        }
        void RemoveBtn_Click(Object Src, EventArgs E) {
            if (InstalledFonts.SelectedIndex != -1) {
                AvailableFonts.Items.Add(new ListItem(InstalledFonts.SelectedItem.Value));
                InstalledFonts.Items.Remove(InstalledFonts.SelectedItem.Value);
            }
        }
        void RemoveAllBtn_Click(Object Src, EventArgs E) {
            while (InstalledFonts.Items.Count != 0) {
                AvailableFonts.Items.Add(new ListItem(InstalledFonts.Items[0].Value));
                InstalledFonts.Items.Remove(InstalledFonts.Items[0].Value);
            }
        }
    </script>
    <body>
       <h3><font face="宋体">处理多个控件操作事件</font></h3>
```

```
<p>
此示例说明如何处理从不同的 &lt;asp:button&gt; 控件引发的多个控件操作事件。
<p>
<hr>
<form id="Form1" action="sample1.aspx" runat=server>
    <table>
        <tr>
          <td>
            可用的字体
          </td>
          <td>
            <!-- Filler -->
          </td>
          <td>
            已安装的字体
          </td>
        </tr>
        <tr>
          <td>
            <asp:listbox id="AvailableFonts" width="100px" runat=server>
                <asp:listitem>Roman</asp:listitem>
                <asp:listitem>Arial Black</asp:listitem>
                <asp:listitem>Garamond</asp:listitem>
                <asp:listitem>Somona</asp:listitem>
                <asp:listitem>Symbol</asp:listitem>
            </asp:listbox>
          </td>
          <td>
            <!-- Filler -->
          </td>
          <td>
            <asp:listbox id="InstalledFonts" width="100px" runat=server>
                <asp:listitem>Times</asp:listitem>
                <asp:listitem>Helvetica</asp:listitem>
                <asp:listitem>宋体</asp:listitem>
            </asp:listbox>
          </td>
        </tr>
        <tr>
          <td>
            <!-- Filler -->
          </td>
          <td>
            <asp:button ID="Button1" text="<<" OnClick="RemoveAllBtn_Click" runat=server/>
            <asp:button ID="Button2" text="<" OnClick="RemoveBtn_Click" runat=server/>
            <asp:button ID="Button3" text=">" OnClick="AddBtn_Click" runat=server/>
            <asp:button ID="Button4" text=">>" OnClick="AddAllBtn_Click" runat=server/>
          </td>
          <td>
            <!-- Filler -->
          </td>
        </tr>
```

```
            </table>
        </form>
    </body>
</html>
```

事件处理程序为开发人员提供了在 ASP.NET 中构造逻辑的清晰方法。例如，上面的示例说明如何在单个页面上连接和处理 4 个按钮事件。该实例运行后的浏览器显示如图 7-2 所示。

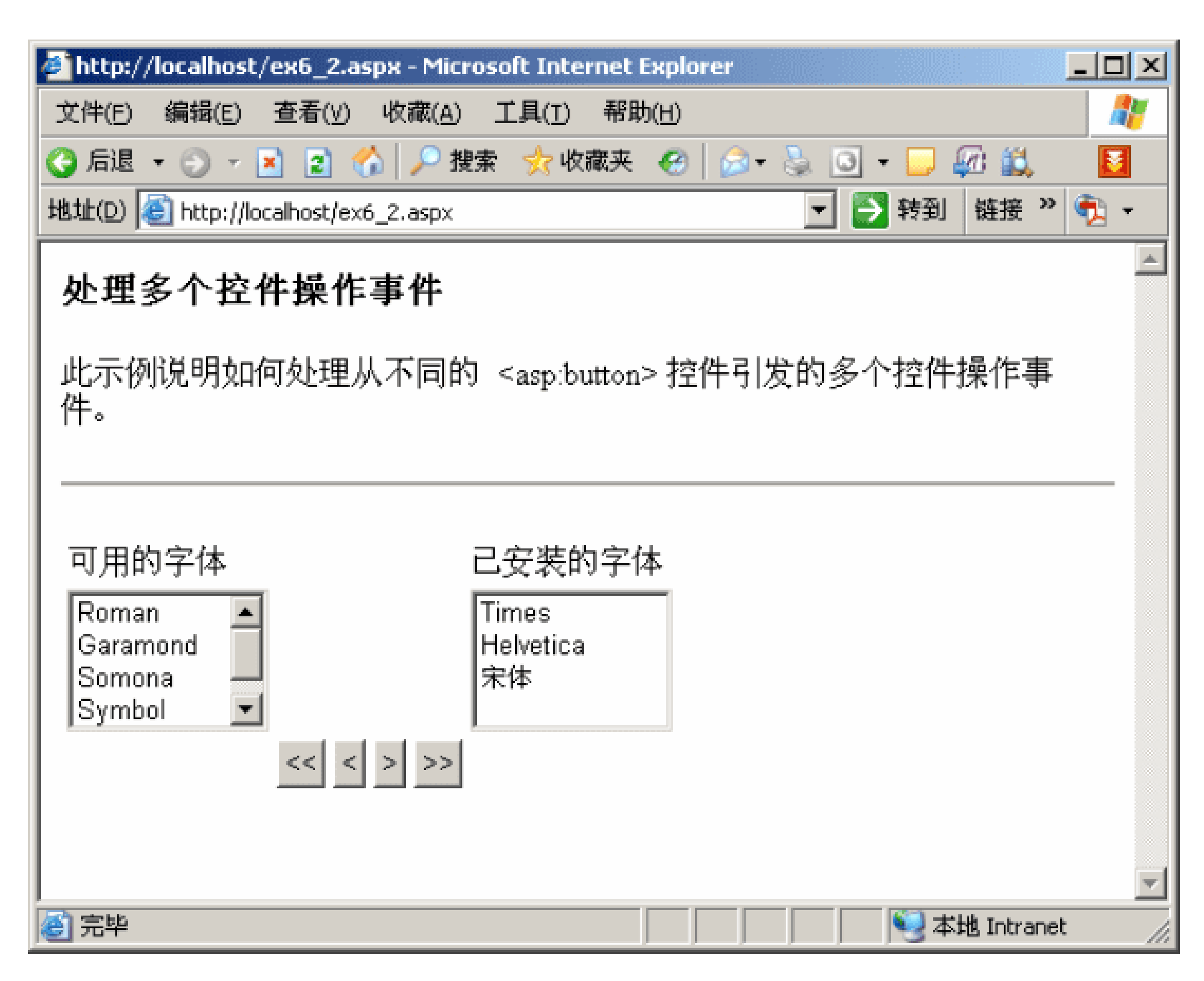


图 7-2　处理多个控件操作事件

注意：

按钮的事件响应方法无须手动添加，在设计界面中，双击按钮，VS 会自动为开发者添加按钮单击方法。其他界面元素组件也类似。

【实例 7-3】到 DataView 的数据绑定

程序代码如 ex7_3.aspx 所示。

ex7_3.aspx

```
<%@ Page Language="C#" %>
<%@ Import namespace="System.Data" %>
<html>
<head>
    <script language="C#" runat="server">
        void Page_Load(Object sender, EventArgs e) {
            if (!Page.IsPostBack) {
                DataTable dt = new DataTable();
                DataRow dr;
                dt.Columns.Add(new DataColumn("整数值", typeof(Int32)));
                dt.Columns.Add(new DataColumn("字符串值", typeof(string)));
                dt.Columns.Add(new DataColumn("日期时间值", typeof(DateTime)));
                dt.Columns.Add(new DataColumn("布尔值", typeof(bool)));
                for (int i = 1; i <= 9; i++) {
                    dr = dt.NewRow();
                    dr[0] = i;
                    dr[1] = "项 " + i.ToString();
```

```
                        dr[2] = DateTime.Now;
                        dr[3] = (i % 2 != 0) ? true : false;
                        dt.Rows.Add(dr);
                    }
                    dataGrid1.DataSource = new DataView(dt);
                    dataGrid1.DataBind();
                }
            }
        </script>
    </head>
    <body>
        <h3><font face="宋体">到 DataView 的数据绑定</font></h3>
        <form id="Form1" runat=server>
            <asp:DataGrid id="dataGrid1" runat="server"
              BorderColor="black"
              BorderWidth="1"
              GridLines="Both"
              CellPadding="3"
              CellSpacing="0"
              HeaderStyle-BackColor="#aaaadd"
            />
        </form>
    </body>
    </html>
```

像 DataGrid、ListBox 和 HTMLSelect 这样的列表服务器控件将集合用作数据源。上面的实例说明如何绑定到通常的公共语言运行库集合类型。这些控件只能绑定到支持 IEnumerable、ICollection 或 IListSource 接口的集合。最常见的是绑定到 ArrayList、Hashtable、DataView 和 DataReader，此处列举了到 DataView 的数据绑定。该实例运行后的浏览器显示如图 7-3 所示。

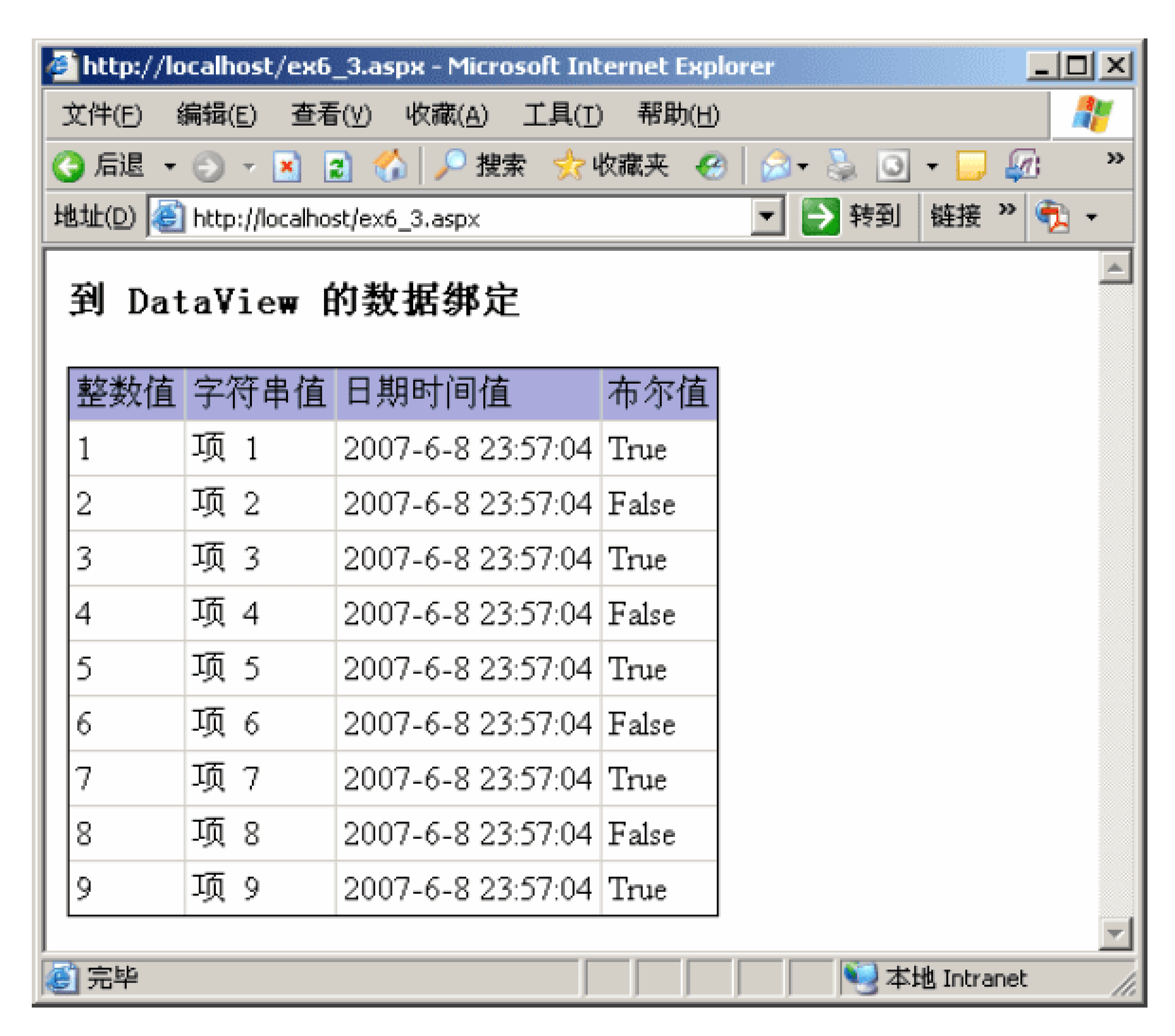


图 7-3　到 DataView 的数据绑定

注意：

在 Visual Studio 中，提供了很多 DataGrid 的设计配色方案，开发者在开发简单的程序时只需从已有的方案中选取合适的即可使页面变得更美观。

【实例 7-4】使用数据库显示数据

程序代码如 ex7_4.aspx 所示。

ex7_4.aspx

```
<% @ Page Language="C#" %>
<%@ Import Namespace="System.Data" %>
<%@ Import Namespace="System.Data.OleDb" %>
<script runat="server">
void Page_Load(Object sender, EventArgs e) {
    OleDbConnection Conn=new OleDbConnection();
    Conn.ConnectionString="Provider=Microsoft.Jet.OLEDB.4.0;"+
                            "Data Source="+Server.MapPath("person.mdb");
    Conn.Open();
    OleDbCommand Comm=new OleDbCommand("select * from grade",Conn);
    OleDbDataReader dr=Comm.ExecuteReader();
    dg.DataSource=dr;
    dg.DataBind();
    Conn.Close();
}
</script>
<asp:DataGrid id="dg" runat="server" />
```

在页面上放置一个 DataGrid 控件，系统自动给出该控件名为 dg。在页面的 Page_Load 过程中首先使用 OleDbConnection 类建立数据库连接对象 Conn，其中的 ConnectionString 属性用于设置连接数据库的相关信息；通过调用其 Open 方法可以打开数据库连接；使用完毕后调用 Close 方法来关闭与数据库的连接；应用 OleDbCommand 类建立 Comm 对象来实现对数据库的操作，查询 person 数据库(见配套源代码中的 person.mdb 文件)grade 表中的数据，并将所有字段的内容显示在屏幕上；通过调用 Comm 对象的 ExecuteReader 方法创建 OleDbDataReader 类的对象 dr；最后将 dg 和数据绑定，实现数据的显示。该实例运行后的浏览器显示如图 7-4 所示。

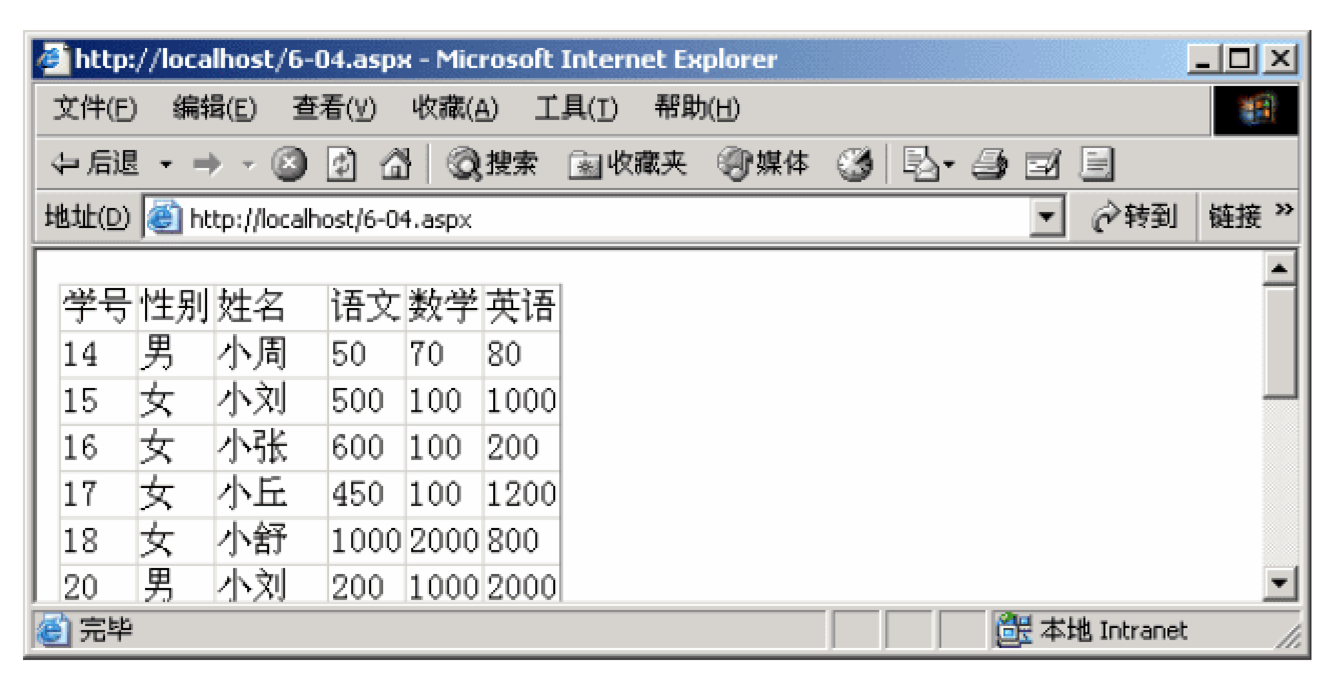


图 7-4　使用数据库显示数据

7.4　Java 技术

随着 Internet 的发展，客户机/服务器计算方面的许多新技术应运而生，其中最为瞩目

的就是 Java。Java 不单定义了一种计算机语言，而且提供了一整套的解决方案，在这个方案中，程序可以按不同的方式发布和运行，也提供了诸如自动下载到客户端并执行的方式。

7.4.1　Java 技术概述

1991 年，当时的 SUN MicroSystem 公司(现被 Oracle 公司收购)的 JameGosling、BillJoe 等人，为在电视、控制烤面包箱等家用消费类电子产品上提供交互操作而开发了一个名为 Oak 的软件，但它当时并没有引起人们广泛的注意。直到 1994 年下半年，Internet 迅猛发展、全球互联网出现了快速增长后，需求促进了 Java 语言的进步，才使得它逐渐成为 Internet 上最受欢迎的开发与编程语言之一。一些著名的计算机公司纷纷购买了 Java 语言的使用权，如当时的 Microsoft、IBM、Novell、Apple、DEC 和 SGI 等，因此 Java 语言被美国著名杂志 *PC Magazine* 评为当年十大优秀科技产品(计算机类仅此一项入选)，随之大量出现了用 Java 编写的软件产品，受到业界的重视与好评。

Java 是一个广泛使用的网络编程语言。首先，作为一种程序设计语言，它简单、面向对象、不依赖于计算机的结构，具有可移植性、健壮性、安全性，并且提供了并发的机制，具有很高的性能。其次，它最大限度地利用了网络，Java 的小应用程序(Applet)可在网络上传输而不受 CPU 和环境的限制。另外，Java 还提供了丰富的类库，使程序设计者可以很方便地建立自己的系统。Java 语言具有以下特点：简单性、面向对象、分布式、解释执行、健壮性、安全性、体系结构中立、可移植性、高性能、多线程以及动态性。

1. 简单性

Java 语言是一种面向对象的语言，它通过提供最基本的方法来完成指定的任务，只需理解一些基本的概念，就可以用它编写出适合于各种情况的应用程序。Java 略去了运算符重载、多重继承等模糊的概念，并且通过实现自动垃圾收集大大简化了程序设计者的内存管理工作。另外，Java 也适合于在小型机上运行，它的基本解释器及类的支持只有 40kB 左右，加上标准类库和线程的支持也只有 215kB 左右。库和线程的支持也只有 215kB 左右。

2. 面向对象

Java 语言的设计集中于对象及其接口，它提供了简单的类机制以及动态的接口模型。对象中封装了它的状态变量以及相应的方法，实现了模块化和信息隐藏。而类则提供了一类对象的原型，并且通过继承机制，子类可以使用父类所提供的方法，实现了代码的复用。

3. 分布式

Java 是面向网络的语言。通过它提供的类库可以处理 TCP/IP 协议，用户可以通过 URL 地址在网络上很方便地访问其他对象。

4. 健壮性

Java 在编译和运行程序时，都要对可能出现的问题进行检查，以消除错误的产生。它提供自动垃圾收集来进行内存管理，防止程序员在管理内存时容易产生的错误。通过集成

的面向对象的例外处理机制，在编译时，Java 提示可能出现但未被处理的例外，帮助程序员正确地进行选择以防止系统的崩溃。另外，Java 在编译时还可捕获类型声明中的许多常见错误，防止动态运行时不匹配问题的出现。

5. 安全性

用于网络、分布环境下的 Java 必须要防止病毒的入侵。Java 不支持指针，一切对内存的访问都必须通过对象的实例变量来实现，这样就防止了程序员使用“特洛伊”木马等欺骗手段访问对象的私有成员，同时也避免了指针操作中容易产生的错误。

6. 体系结构中立

Java 解释器生成与体系结构无关的字节码指令，只要安装了 Java 运行系统，Java 程序就可在任意的处理器上运行。这些字节码指令对应于 Java 虚拟机中的表示，Java 解释器得到字节码后，对它进行转换，使之能够在不同的平台运行。

7. 可移植性

与平台无关的特性使 Java 程序可以方便地被移植到网络上的不同计算机上。同时，Java 的类库中也实现了与不同平台的接口，使这些类库可以移植。另外，Java 编译器是由 Java 语言实现的，Java 运行时系统由标准 C 实现，这使得 Java 系统本身也具有可移植性。

8. 解释执行

Java 解释器直接对 Java 字节码进行解释执行。字节码本身携带了许多编译时的信息，使得连接过程更加简单。

9. 高性能

和其他解释执行的语言如 BASIC、TCL 不同，Java 字节码的设计使之能很容易地直接转换成对应于特定 CPU 的程序代码，从而得到较高的性能。

10. 多线程

多线程机制使应用程序能够并行执行，而且同步机制保证了对共享数据的正确操作。通过使用多线程，程序设计者可以分别用不同的线程完成特定的行为，而不需要采用全局的事件循环机制，这样就很容易地实现了网络上的实时交互行为。

11. 动态性

Java 的设计使它适合于一个不断发展的环境。在类库中可以自由地加入新的方法和实例变量而不会影响用户程序的执行。并且 Java 通过接口来支持多重继承，使之比严格的类继承具有更灵活的方式和扩展性。

7.4.2　Applet 与 Application

在客户端 Java 程序具有两种不同类型：小应用程序(Applet)和应用程序(Application)，

Applet 是嵌入 Web 文档的程序，而 Application 则是一般的应用程序。因为 Applet 主要是用于网络通信，由于传输速度有限，如果较大则可能下载时间较长，因而 Applet 一般来说规模较小，而对于 Application 则无此顾虑。

Applet 与 Application 之间的技术差别来源于其运行环境的差别。Applet 需要来自浏览器的大量信息：浏览器客户机的位置和大小、嵌入主 HTML 文档的参数、初始化过程(init)、启动过程(start)、停止过程(stop)、终止过程(destory)、绘图过程(paint)等。而 Application 则相对要简单得多，它来自外部的唯一输入就是命令行参数。

以下是一个 Java 程序，它既能作为 Applet 又能作为 Application。

【实例 7-5】同时在网页及 Windows 窗体中显示 Hello World

程序代码如 ex7_5.java 所示。

ex7_5.java

```
import javax.swing.*;
import java.awt.*;
import java.applet.Applet;
import java.awt.event.*;
public class ex7_5 extends Applet{          // 第 5 行
    public static void main(String args[]){
        JFrame frame=new JFrame("Application");
        ex7_5 app = new ex7_5();
        frame.getContentPane().add(app,BorderLayout.CENTER);
        frame.setSize(150,100);
        frame.setVisible(true);
        frame.addWindowListener(new WindowControl(app));
        app.init();
        app.start();
    }
    public void paint(Graphics g){
        g.drawString("Hello,World!",25,25);
        g.drawRect(20,10,80,20);
    }
    public void destroy(){
        System.exit(0);
    }
}
class WindowControl extends WindowAdapter{
    Applet c;
    public WindowControl(Applet c){
        this.c=c;
    }
    public void WindowControl(WindowEvent e){
        c.destroy();
    }
}
```

ex7_5.html

```
<!doctype html public "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
  <head><title>ex7_5</title>
  </head>
  <body bgcolor="000000">
    <Applet code = "ex7_5.class" width=300 height=200>
    </Applet>
  </body>
</html>
```

假设文件 ex7_5.java 和 ex7_5.html 存放在 C:\SRC 目录下，则运行该程序的步骤如下：

(1) 首先需要安装 JDK，安装及配置方法可参考相关文档说明。

(2) 在“开始”菜单中单击“运行”或直接用键盘的“Windows 键+R”打开“运行”对话框，在此对话框中输入 cmd 并按 Enter 键。

(3) 在弹出的 DOS 窗口中，输入命令“cd \SRC”，再按 Enter 键，将当前目录切换为存放源代码的目录，此时可观察到提示符变为“C:\SRC>”。

(4) 输入命令“javac ex7_5.java”，再按 Enter 键，稍等片刻后又回到命令行方式。如果出现提示信息，则说明编译出现了所提示的问题。如果编译成功，则可用“dir”命令检查是否自动生成了 ex7_5.class 和 Windowcontrol.class 文件。

(5) 输入命令“java ex7_5”，再按 Enter 键，此时会在桌面的左上角出现如图 7-5 所示的窗口，其中显示了“Hello,World!”。

(6) 在“资源管理器”中双击 ex7_5.html 文件，可以看到如图 7-6 所示的浏览器中显示的“Hello,World!”。

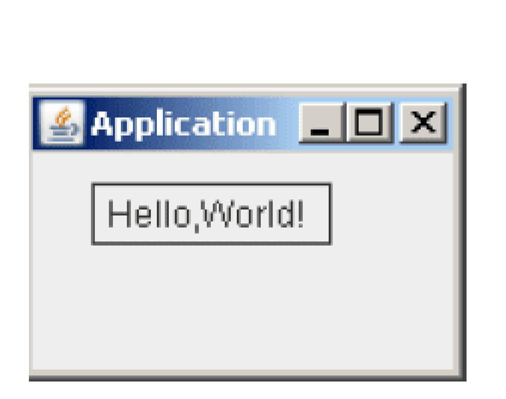


图 7-5　应用程序

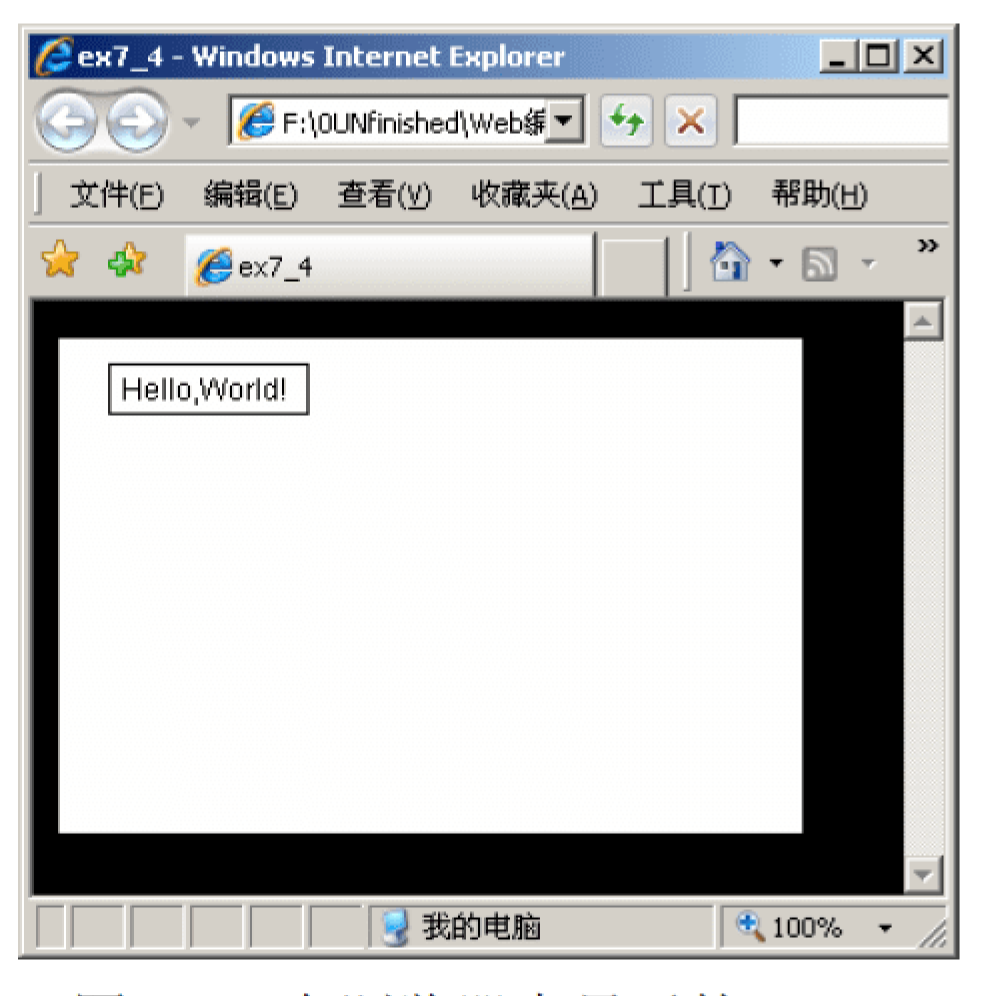


图 7-6　在浏览器中显示的 Applet

注意：

在网站正式运行时，只需要将扩展名为.html 和.class 的文件发布到服务器，.java 文件没有必要发布上去，以免代码泄露。

本程序的作用是在屏幕上输出：Hello,World!。本程序特殊之处在于编译后本程序可以

同时在网页中和 Windows 窗体中显示。首先，Applet 必须作为 java.applet.Applet 的子类，而 Application 则必须有一个公共的方法 main()。其次，两者的主线程是不同的，Applet 是由方法 init()对 Applet 进行初始化的，而 Application 则从方法 main()开始运行程序。Applet 由浏览器管理其生命周期(Life Cycle)，即生成(new)、初始化(init)、运行(start)、停止(stop)和销毁(destroy)等；而 Application 则自行管理其生命周期。一般而言，Java 的 Applet 和 Application 是完全遵照前面的某一种方式进行编写的，但 Java 允许写出既是 Applet 又是 Application 的程序。这样，既可以进一步了解 Java 的内部结构又可以使同一程序运行于不同的运行环境。

在程序代码 ex7_5.java 中，第 5 行的 extends Applet，表示程序继承 java.applet.Applet 类。在类中，重写了父类 Applet 的 paint()方法，其中参数 g 为 Graphics 类，它表明当前画板的上下文。在 paint()方法中，调用 g 的方法 drawString()，在坐标(25,25)处输出字符串，再调用 drawRect()方法画一个矩形。如果作为 Application，则由 main()方法开始，先生成程序本身的实例将程序加入窗口，然后调用 init()方法进行初始化。

注意：

并不是所有的小应用程序都可能同时也是应用程序，因为有一些在小应用程序中的功能不能用于应用程序中，如 Applet.getCodeBase()、Applet.getDocumentBase() 等在 Application 中应用时就会抛异常。而一些在 Application 中可以使用的程序，由于安全问题，也不能在 Applet 中使用，毕竟 Applet 是要发布在网上的，需要更高的安全性。

近几年 Flash 等 RIA 技术发展比较迅速，从界面的友好性、下载速度等很多方面都超过了 Java Applet，并且大有取代 Java Applet 之势。但是在某些方面 Java Applet 还是具有优势的，比如，处理复杂数据和实现复杂的逻辑等。

7.4.3　Servlet

可以将 Servlet 作为服务器端的 Applet。它从客户端接收请求，执行设定的操作后，最终将结果返回给客户端。使用 Servlet 的基本流程如下：

- 客户端(很可能是 Web 浏览器)通过 HTTP 提出请求。
- Web 服务器接收该请求并将其发给 Servlet。如果这个 Servlet 尚未被加载，Web 服务器将把它加载到 Java 虚拟机并且执行它。
- Servlet 将接收该 HTTP 请求并执行某种处理。
- Servlet 将向 Web 服务器返回应答。
- Web 服务器将从 Servlet 收到的应答发送给客户端。

由于 Servlet 是在服务器上执行，不存在类似于 Applet 的安全性问题，所以一些很难由 Applet 实现的功能可以利用 Servlet 并通过 CORBA、RMI、socket 和本地(native)调用的通信等方式来实现。

注意：

Web 浏览器并不直接和 Servlet 通信，Servlet 是由 Web 服务器加载和执行的。这意味

着如果 Web 服务器有防火墙保护，那么其上发布的 Servlet 也将得到防火墙的保护。

由于具有平台无关性，Servlet 可以很好地取代 CGI 脚本，此外 Servlet 还具有如下的特点：

1. 持久性

Servlet 只需 Web 服务器加载一次，而且可以在不同请求之间保持服务(如一次数据库连接)。与之相反，CGI 脚本是短暂的、瞬态的。每一次对 CGI 脚本的请求，都会使 Web 服务器加载并执行该脚本。一旦这个 CGI 脚本运行结束，它就会被从内存中清除，然后将结果返回到客户端。CGI 脚本的每一次使用，都会造成程序初始化过程(如连接数据库)的重复执行。

2. 与平台无关

Servlet 是用 Java 编写的，它自然也继承了 Java 的平台无关性。

3. 可扩展性

由于 Servlet 是用 Java 编写的，它就具备了 Java 所能带来的所有优点。Java 是健壮的、面向对象的编程语言，它很容易扩展以适应用户的需求，Servlet 自然也继承了这些特征。

4. 安全性

从外界调用一个 Servlet 的唯一方法就是通过 Web 服务器。这提供了高水平的安全性保障，尤其是在用户的 Web 服务器有防火墙保护的时候。

5. 可在异构的客户机上使用

由于 Servlet 是用 Java 编写的，所以可以很方便地在 HTML 中使用，就像使用 Applet 一样。

【实例 7-6】一个简单的 Servlet 实例

程序代码如 ex7_6.java 所示。

ex7_6.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
public class ex7_6 extends HttpServlet {
   public void service(HttpServletRequest req,HttpServletResponse res) throws IOException
   {
      res.setContentType("text/html");
      PrintWriter out = res.getWriter();
      out.println("<html><head><title>Hello World!</title></head>");
      out.println("<body>");
      out.println("<h1>Hello World!</h1></body></html>");
   }
}
```

首先在 Tomcat 服务器的 Web 应用主目录(如 C:\Tomcat\webapps\)下建立 servletdemo，并添加一个子目录 WEB-INF，将编译后的 ex7_6.class 文件复制到 WEB-INF 下的 class 子目录下，再编辑 Tomcat 的配置文件 web.xml，以下是该文件中和本程序有关的片段。

web.xml

```
<servlet>
<servlet-name> ex7_6</servlet-name>
<servlet-class>examples.servlets. ex7_6</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name> ex7_6</servlet-name>
<url-pattern>/ex7_6/*</url-pattern>
</servlet-mapping>
```

在浏览器中用 http://localhost:8080/servletdemo/ex7_6 即可完成调用。这个程序用 service()方法实现对客户端的响应。在这个响应中，首先调用了 setContextType("text/html")设置响应内容类型。因为要发送文本，用 getWriter()方法获得了 PrintWriter 对象，调用 out.println()将要发送给客户端的信息逐行生成，本实例运行的结果如图 7-7 所示。

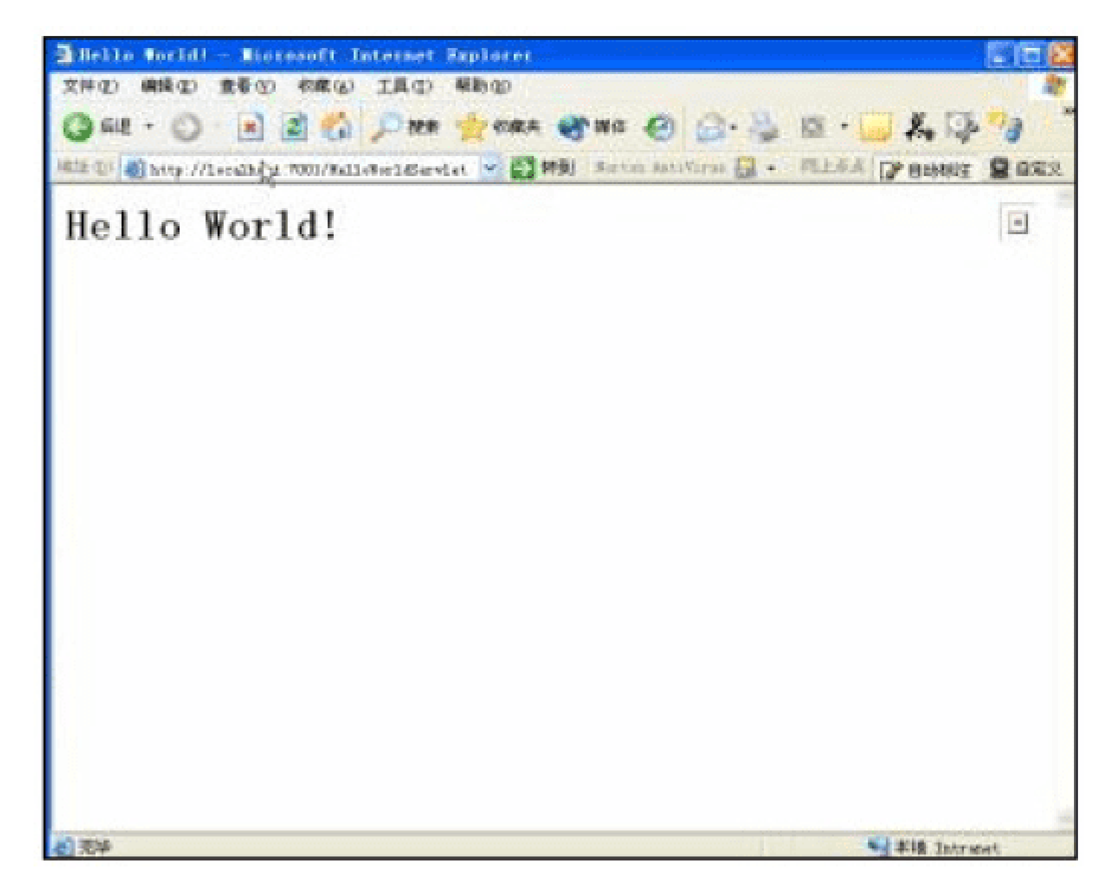


图 7-7　简单的 Servlet 实例

但 Java Servlet 也不是没有缺点，和传统的 CGI、ISAPI/NSAPI 方式相同，Java Servlet 是利用输出 HTML 语句来实现动态网页的，如果用 Java Servlet 来开发整个网站，动态部分和静态页面的整合过程简直就是一场恶梦。这就是为什么 JSP(Java Server Pages)被推出的原因。

7.4.4　JSP

JSP 提供了一种简单而快速的方法创建显示动态生成内容的 Web 页面。JSP 技术规范定义了如何在服务器和 JSP 之间进行交互，描述了页面的格式和语法。它使用 XML 标签和 Scriptlets(一种使用 Java 语言编写的脚本代码)封装生成页面内容的逻辑。JSP 将各种格式的标签(HTML 或 XML)直接传递给响应的页面。通过这种传送方式，JSP 实现了页面逻辑与其设计和显示的分离。按照脚本语言是服务于某一个子系统的语言这种论述，JSP 应当被看作是一种脚本语言，然而，作为一种脚本语言，它又显得过于强大了，在 JSP 中几乎可以使用全部的 Java 类。JSP 在执行时被编译成 Servlet，并可调用 JavaBean 组件或 Enterprise JavaBean 组件，以便在服务器端处理。因此，JSP 技术在构建可升级的基于 Web 的应用时扮演了重要角色。JSP 与 Java 语言一样，并不局限于任何特定的平台或 Web 服务器。因此，如果 Web 服务器没有提供 ASP 支持，比如使用了 Apache 或 Tomcat 等服务器时，可以考虑使用 JSP。

注意：

请不要将 JSP 与服务器端的 JavaScript 混为一谈。网站服务器会自动将以 JSP 写成的 Java 程序代码段转换成 Java Servlets。而许多先前必须以 Perl 手写程序或服务器特定的 API(如 ASP)控制的功能也都可透过 JSP 来自动处理。

1. JSP 的优点

既然 JSP 在执行时也要被编译成 Servlet，那么在理论上就可以直接编写 Servlet 创建 Web 应用。然而，JSP 技术通过将页面内容和显示逻辑分开，简化了创建网页的过程。在许多应用程序中，需要将模板内容和动态生成的数据一块发送到客户端。基于这种考虑，使用 JSP 技术要比全部使用 Servlet 方便得多。JSP 技术具有以下主要优点。

(1) 简单实用

JSP 可以实现大部分的 Servlet 功能，并继承了 Servlet 的优点，弥补了 Servlet 的不足。JSP 使 Servlet 的动态数据处理和输出格式两者分开，采用了一种类似 HTML 的格式，使 Web 应用的维护和修改更加方便。即使不懂 Java 编程的人员也可以通过标准标签实现 JSP 的基本功能。

(2) 移植性和规范性好

JSP 技术和微软公司的 ASP 技术是竞争关系。JSP 使用 Java 语言作为动态内容生成的编程语言，ASP 则主要使用 VBScript 脚本语言。ASP 程序一般运行在微软的 IIS 服务器上，一旦从 Windows 平台转到其他平台，就很难再被使用。Java 语言则具有更强的适用性和移植性，JSP 程序无须改动，就可以在各种平台上运行。

Servlet 和 JSP 是 Java 技术在 Web 层的主要技术。Servlet 是用 Java 语言编写的 Java 类，能动态处理客户请求并构造响应。JSP 则基于文本，也能像 Servlet 一样被执行，更多适用于创建一些静态内容。

2. JSP 开发方式

JSP 既可以用于开发小型的 Web 站点，也可以用于开发大型的、企业级的应用程序，使用 JSP 存在几种不同的开发方式。

(1) 直接使用 JSP

对于最小型的 Web 站点，可以直接使用 JSP 来构建动态网页，这种站点最为简单，所需要的仅仅是简单的留言板、动态日期等基本的功能。对于这种开发模式，一般可以将所有的动态处理部分都放置在 JSP 的 Scriptlet 中。

(2) JSP+JavaBeans

中型站点面对的是数据库查询、用户管理和小量的商业业务逻辑。对于这种站点，不能将所有的东西全部交给 JSP 页面来处理，而是在其中加入 JavaBeans 技术来帮助中型网站的开发。利用 JavaBeans，将很容易完成如数据库连接、用户登录与注销、商业业务逻辑封装的任务。如将常用的数据库连接写为一个 JavaBeans，既方便了使用，又可以使 JSP 文件简单而清晰，通过封装，还可以防止一般的开发人员直接获得数据库的控制权。

(3) JSP+JavaBeans+Servlet

无论用 ASP 还是 PHP 开发动态网站，长期以来都有一个比较重要的问题，就是网站的逻辑关系和网站的显示页面不容易分开，最终的作品也几乎无法阅读。另外，动态 Web 的开发人员也在抱怨，将网站美工设计的静态页面和动态程序合并的过程是一个异常痛苦的过程。

如何解决这个问题呢？在JSP问世以后，人们认为Servlet已经完全可以被JSP代替，然而，事实是当Servlet不再担负动态页面生成的任务以后，开始担负起决定整个网站逻辑流程的任务。在逻辑关系异常复杂的网站中，借助于Servlet和JSP良好的交互关系和JavaBeans的协助，完全可以将网站的整个逻辑结构放在Servlet中，而将动态页面的输出放在JSP页面中来完成。在这种开发方式中，一个网站可以有一个或几个核心的Servlet来处理网站的逻辑，通过调用JSP页面来完成客户端(通常是Web浏览器)的请求。后面我们将可以看到，在J2EE模型中，Servlet的这项功能可以被EJB取代。

【实例 7-7】显示不同颜色的文字

程序代码如 ex7_7.jsp 所示。

ex7_7.jsp

```
<HTML>
<HEAD>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<TITLE>JSP test page---HelloWorld!</TITLE>
</HEAD>
<BODY>
<%
String[] colors={"red","green","blue","black","gray"};
for (int i = 0 ; i < 5 ; i++)
{
        out.println("<h1><font color=" + colors[i] + ">Hello World! My first jsp page.</font></h1>");
}
%>
</BODY>
</HTML>
```

在这个例子中，首先定义了一个包含 5 个字符串的数组，其中每一个成员都定义了一种颜色，在下面的代码中，使用这 5 种颜色循环输出文字，该实例运行后浏览器中的显示如图 7-8 所示。



图 7-8　显示不同颜色的文字

7.4.5　J2EE

J2EE(Java 2 Platform，Enterprise Edition)平台建立在 J2SE(Java 2 Platform，Standard

Edition)的基础上，为企业级应用提供了完整、稳定、安全和快速的 Java 平台，它是一个标准而不是一个产品。J2EE 平台提供的 Web 开发技术主要支持两类软件的开发和应用，一类是做高级信息系统框架的 Web 应用服务器(Web Application Server)，另一类是在 Web 应用服务器上运行的 Web 应用(Web Application)。

J2EE 体系结构如图 7-9 所示。一方面，与最终用户进行交互的前端表示组件在逻辑上被划分到了客户层，而提供数据存储与访问功能的组件被划分到了数据层。另一方面，在逻辑上驻留在前端与后端之间的中间层可能由一个表示逻辑层和一个业务层组成。表示逻辑层包括基于 Internet 协议和 Web 协议(HTTP、HTTPS、HTML 和 XML)提供应用功能的组件，业务层由捕获企业业务逻辑的组件组成。这两个层在逻辑上可划分为完全分离的两层，每一个分离的层都是独立的，从而使 J2EE 支持分布式 4 层(或者 n 层)应用。J2EE 是一个灵活的结构，它不将开发人员锁定到特定数量的层上，并且不详细规定对于这些逻辑分组的物理分离。在网络计算环境中，一个普通的应用可以在一台计算机上同时运行表示逻辑层和业务层(甚至可以包括数据层)，而高级的应用可以在若干台计算机上从物理上分隔每一层。

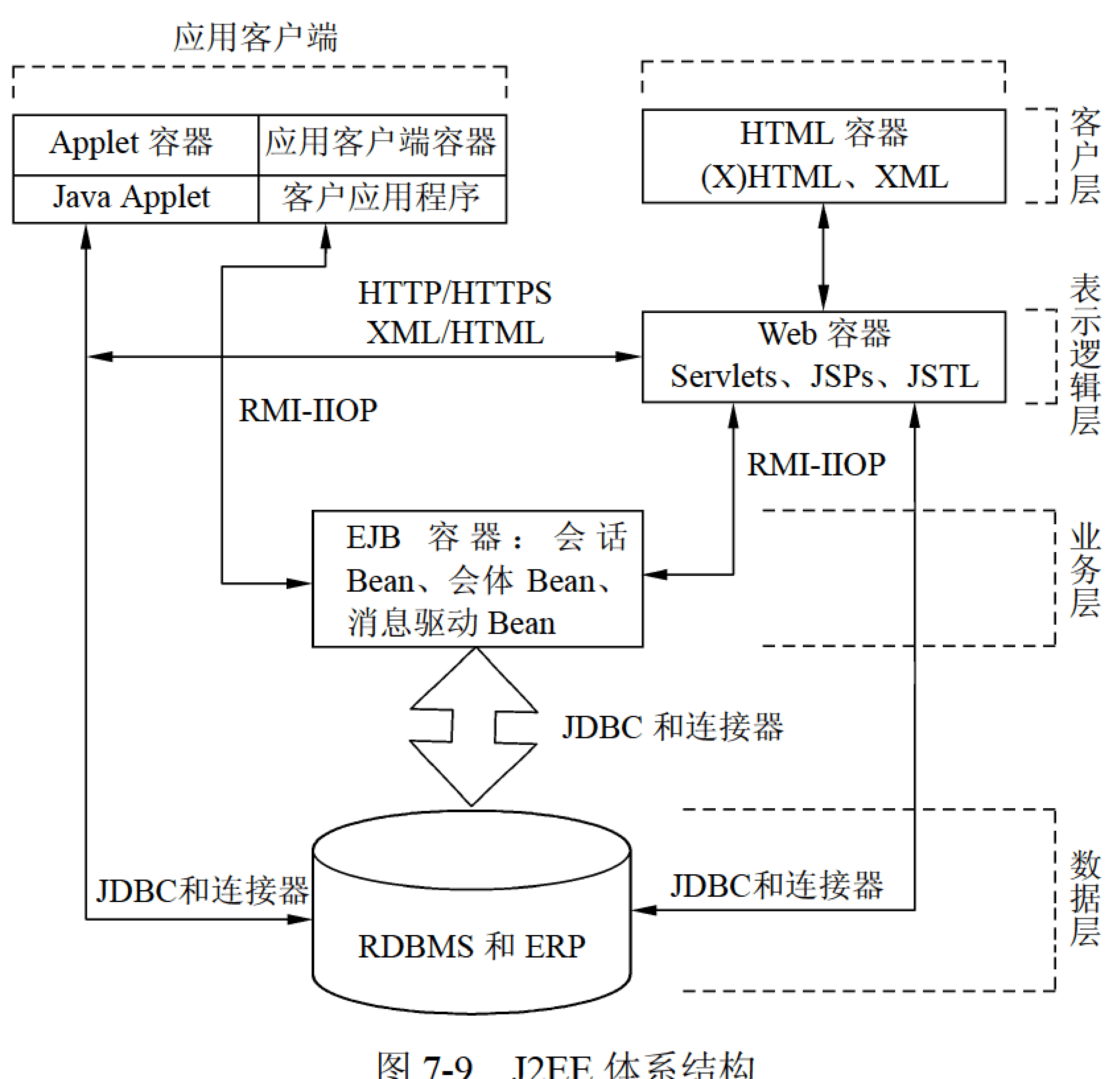


图 7-9　J2EE 体系结构

在 J2EE 的开发中，MVC 是一种非常重要的设计模式，它通常将整个系统分为三个主要部分。

1. 视图

视图就是用户界面部分，在 Web 应用程序中也就是 HTML、XML、JSP 页面。这个部分主要处理用户看到的东西，动态的 JSP 部分处理了用户可以看见的动态网页，而静态的网页则由 HTML、XML 输出。

2. 控制器

控制器负责网站的整个逻辑。它用于管理用户与视图发生的交互。可以将控制器想象成处在视图和数据之间，对视图如何与模型交互进行管理。通过使视图完全独立于控制器和模型，就可以轻松替换前端客户程序，就是说，网页制作人员将可以独立自由地改变 Web 页面而不用担心影响这个基于 Web 的应用程序的功能。

在 J2EE 中，控制器的功能一般是由 Servlet、JavaBeans、EJB(Enterprise Java Beans，即定义了一个用于开发基于组件的企业多重应用程序的标准)中的 SessionBean 来担当的。

3. 模型

模型就是应用业务逻辑部分，这一部分的主角是 EJB，借助于该强大的组件技术和企业级的管理控制，开发人员可以轻松形创建出可重用的业务逻辑模块。

注意：

J2EE 较为复杂，此处仅对它做了非常简要的介绍，如果希望获取更详细的资料，可参考专门介绍 J2EE 方面的书籍。

由于 J2EE 这个平台没有能够提供一个令人满意的应用程序编程模型(Application Programming Model)。一些大的应用服务器供应商试图用开发工具来降低 J2EE 开发的复杂性，但是很多 J2EE 开发工具自动产生的代码像这些工具本身一样复杂。因此很多开发者选择了另外一些开发方式，如 Struts、Hibernate 和 Spring Framework 等，它们都能有效降低 J2EE 开发难度的开发框架，使用它们可以大大加快开发的速度。

7.5　不同的动态网页技术比较

7.5.1　CGI

CGI(Common Gateway Interface)，即通用网关接口的简写，它是一个 Web 服务器主机提供信息服务的标准接口，通过提供这样一个标准接口，Web 服务器能够执行应用程序并将它们的输出，如文字、图形和声音等传递给一个 Web 浏览器。

一般来说，CGI 标准接口的功能就是在超文本文档与服务器应用程序之间传递信息。如果没有 CGI，Web 服务器只能提供静态文本或者连接到其他服务器。可以毫不夸张地说，有了 CGI，万维网才变得更为实用，界面才变得更为友好，信息服务才变得更为丰富多彩。

CGI 是一个连接外部应用程序到信息服务器(比如 HTTP 或者网络服务器)的标准。一个简单的 HTML 文档无交互后台程序，它是静态的，也就是说它处于一个不可变的状态，即文本文件不可以变化。相反地，CGI 程序是可以实时执行的，它可以输出动态的信息。CGI 技术原理图如图 7-10 所示。

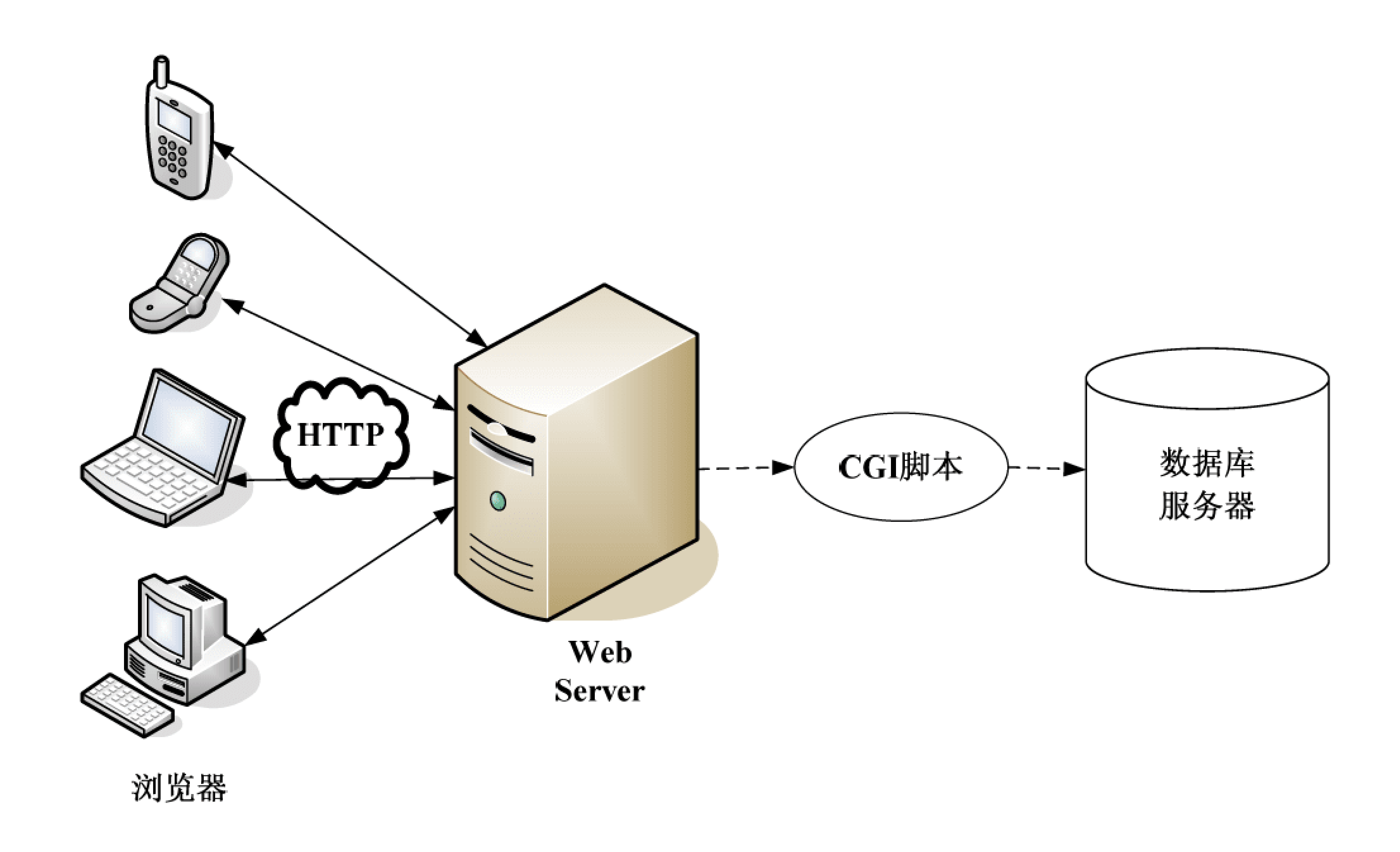


图 7-10　CGI 技术原理图

CGI 工作的主要流程是：

(1) 首先，一个用户请求激活一个 CGI 应用程序。

(2) CGI 应用程序将交互主页中用户输入的信息提取出来。

(3) 将用户输入的信息传给服务器主机应用程序(如数据库查询)。

(4) 将服务器处理结果通过 HTML 文件返回给用户。

(5) CGI 进程结束。

CGI 程序的工作原理是：客户端的 Web 浏览器浏览到某个主页后，利用一定的方式提交数据，并通过 HTTP 协议向 Web 服务器发出请求，服务器端的 HTTP 守护进程将描述的主页信息通过标准输入 stdin 和环境变量(environment variable)传递给主页指定的 CGI 程序，并启动此应用程序进行处理(包括对数据库的处理)。处理结果通过标准输出 stdout 返回给 HTTP 守护进程，再由 HTTP 守护进程通过 HTTP 协议返回给客户端的浏览器，由浏览器负责解释执行，将最终的结果显示给用户。

由 CGI 的运行原理可知，只要能进行标准输入和标准输出的编程工具都可以用于编写 CGI 程序，因此程序几乎可以用任何语言来编写(C/C++、PERL、Java 和 Visual Basic 等)，并且可以在不同平台(Windows/Linux 等)中执行。实际上，从广义上说，PHP、ASP 等也可以是 CGI 脚本语言。

下面是一个简单的 Perl 程序：

```
#! /usr/bin/perl
print "你好！ 欢迎学习 Web 开发基础! \n";
```

这里的第一行说明了这是一个 Perl 程序，它也是 Perl 的注释，注释是从#开始至该行结束的所有文字。第二行是程序的可执行部分，此处只有一条 print 语句，如果学过 C 或 C++语言，这门语言是很容易掌握的。

CGI 程序一般是可执行程序。编译好的 CGI 程序一般要集中放在一个目录下。具体存放的位置随操作系统的不同而不同(而且可以由用户自己根据情况进行配置)，例如 UNIX

系统下的 WWW 服务器中一般放在 cgi-bin 子目录下，而在 Windows 操作系统下以 Webstar 或 Website 作 WWW 服务器，CGI 程序都放在 cgi-win 下。不过，CGI 程序的执行一般有两种调用方式：一是通过 URL 直接调用， 如“http://202.205.240.63/cgi－bin/test.cgi”，在浏览器的 URL 栏里直接输入上述地址就可以调用该程序；另一种方式，也是主要的方式，是通过交互式主页中的 FORM 栏调用，通常都是用户在填完一张输入信息主页后按“确认”按钮启动 CGI 程序。主页的交互一般都是通过这样调用 CGI 来完成的。

7.5.2　ISAPI/NSAPI

CGI 程序每接收到一个来自于浏览器的请求就需要在系统中创建一个新的进程，从磁盘上装载可执行映像，并在完成时再把它全部清除。另外，在每一次调用时有些资源(如数据库连接资源)必须重新建立，它们既不能被缓存也不能被重用。这样随着网站访问人数的增加，Web 服务器的性能也必将直线下降，导致服务功能下降或不能提供服务。另外，它的数据库连接功能比较弱，某些情况下甚至还会导致数据库服务系统的存取速度下降。

由于 CGI 存在上述缺点，为了改善性能人们开始尝试采用 API(Application Program Interface)接口调用方式来实现相同的功能。ISAPI/NSAPI(Internet/Netscape Server Application Programming Interface)网关程序的最大优点在于当它在服务器端第一次被执行的时候即被调入内存，在本次请求结束后也不退出。这样做的好处在于：无论浏览器传送过来多少个请求，在服务器端都由同一个进程进行响应，执行效率高且能保持与数据库之间的高效连接。但是服务器必须同时启动多个线程来处理多台浏览器同时对一个 ISAPI/NSAPI 网关程序的进程发出请求，这就加大了开发的难度。

从开发的角度，ISAPI/NSAPI 网关程序可以使用各种高级语言进行开发，如 Visual C++、DELPHI 等。编译后会得到一个 DLL(Windows 平台)形态的软件，对于程序员来说，直接使用“应用编程接口”意味着高度可控，但调试困难则意味着编写不易。

ISAPI 编程与目前流行的其他 Web 开发方式比较，其优势主要表现在性能、安全、功能等方面。有权威机构做过评测，ISAPI 在各项指标上领先于 NSAPI，与目前被大量使用的 Web 开发脚本语言，如 ASP、PHP、JSP 等比较，其运行效率较高。

7.5.3　ASP

ASP(Active Server Page 动态服务器页面，简称 ASP)是一套微软开发的服务器端脚本平台，ASP 内含于 IIS 当中。

通过 ASP 可以结合 HTML 网页、ASP 指令和 ActiveX 组件建立动态、交互且高效的 Web 服务器应用程序。同时，它也支持 VBScript 和 JavaScript 等脚本语言，默认为 VBScript。

ASP 采取服务器解析之后再向浏览器返回数据的方式来生成网页，所以有了 ASP 就不必担心客户的浏览器是否能运行程序员所编写的代码。因为所有的程序都将在服务器端执行，包括所有嵌在普通 HTML 中的脚本程序。当程序执行完毕后，服务器仅将执行的结果返回给客户浏览器，这样也就减轻了客户端浏览器的负担，大大提高了交互的效率。

但是这种方式也导致一个潜在的问题，即运行 ASP 页面比普通的 HTML 页面要慢一

些。这是因为普通的 HTML 页面只需要浏览器就能够解析，而 ASP 则必须是服务器将整页的代码都执行完毕之后才能返回数据。也正是这个机制，在客户端看到的只能是经过解析之后的数据，而无法获得源代码，故程序员不用担心自己的代码会被别人剽窃。

1. ASP 的特点

(1) 无须编译：容易产生，无须编译或链接即可执行，集成于 HTML 中。使用常规文本编辑器，如记事本即可编辑。

(2) 与浏览器无关：客户端只要使用常规的浏览器即可浏览，所设计的 ASP 代码在站点服务器端执行。

(3) 面向对象：可通过 ActiveX Server Components(ActiveX 服务器组件)来扩充功能。而 ActiveX 服务器组件可使用 Visual Basic、Java、Visual C++、COBOL 等多种语言来编写。

(4) 与任何 ActiveX Scripting 语言兼容：除了可使用 VBScript 或 JScript 语言来设计，并可通过 Plug-in 的方式，使用由第三方所提供的其他譬如 REXX、perl、Tcl 等脚本语言。

(5) ASP 脚本服务器解析：可以保护所创作的源程序不外泄。客户浏览器一侧只能看到 ASP 脚本执行生成之后的常规 HTML 代码。

2. ASP 的对象

Active Server Pages 提供了 5 个可以直接调用的内置的“对象”(object)，分别如下。

(1) Request：取得用户信息；

(2) Response：传送信息给用户；

(3) Server：提供访问服务器的方法(methods)和属性(properties)的功能；

(4) Application：一个应用程序，可以在多个主页之间保留和使用一些共同的信息；

(5) Session：一个用户，可以在多个主页之间保留和使用一些共同的信息，在多个主页之间共享信息。

3. ASP 页面间的参数传递

ASP 开发的应用程序，可以在多个主页之间保留和使用一些共同的信息，ASP 提供两种适用范围分别如下。

(1) Application：应用的所有信息，在一个应用程序、多个主页间，实现所有用户共同信息的共享和使用。

(2) Session：会话的所有信息，仅适用于一个用户的会话。

4. ASP 的使用

Active Server Pages(ASP)制作成.ASP 扩展名的文件，一个.ASP 文件是一个文本文件，包括 HTML 标签(tags)，VBscript 或 JavaScript 语言的程序码，ASP 的语法。

ASP 并不是一个脚本语言，而是提供一个可以集成 script 语言(VBscript 或 JavaScript)到 HTML 主页的环境。HTML 标签(tags)使用“<..>”将 HTML 程序码包含起来，以与常规的文本区分开来；而 ASP 则使用“<%...%>”将 ASP 程序码包含起来。

【实例 7-8】显示现在的日期时间

程序代码如 ex7_8.asp 所示。

ex7_8.asp

```
<html>
现在是：<%=Now%>
</html>
```

该程序运行后显示当前的日期时间，其运行结果如图 7-11 所示。

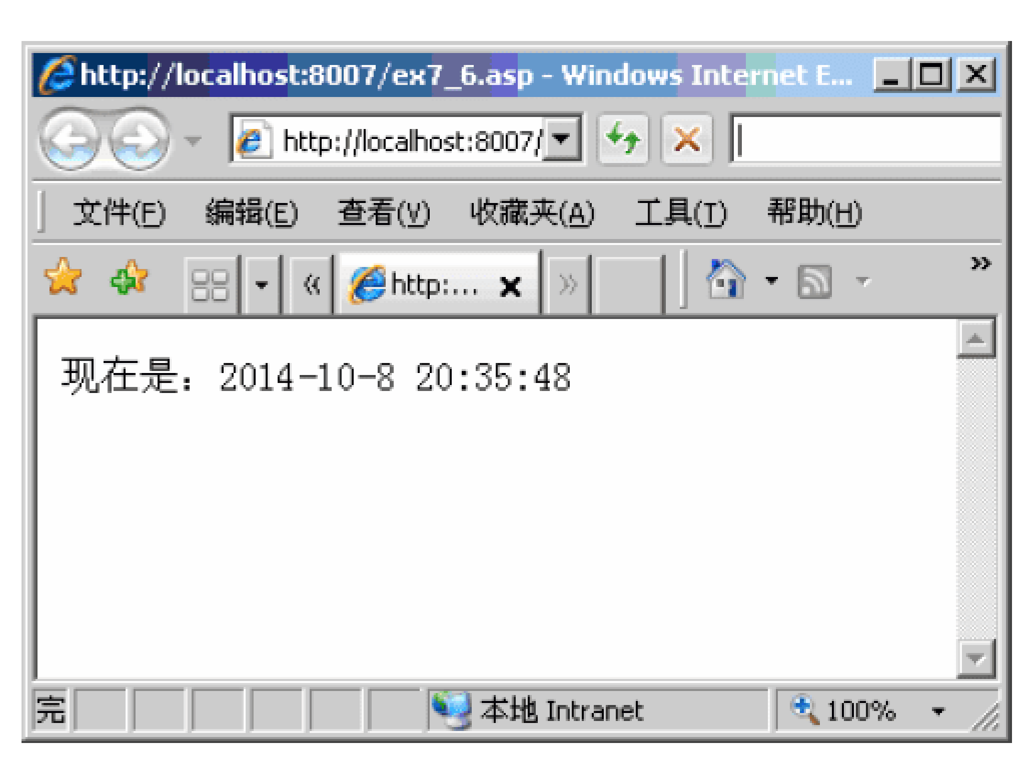


图 7-11　显示现在的日期时间

注意：

ASP 目前只能运行在微软的 Windows 平台上。在某些版本的 Windows 系统中，有时 IIS 在 Windows 初始安装时未被选中，且部分系统在 IIS 默认安装后没有配置 ASP 的支持，因此必须保证 IIS 的正确安装和配置后才能正常运行 ASP。

【实例 7-9】重复循环显示

程序代码如 ex7_9.asp 所示。

ex7_9.asp

```
<html>
  <%for i = 1 to 7 %>
    <font size= <% = i %>color=#0000ff>掌握动态网页技术，未来在我脚下！
    </font><br>
  <%next%>
</html>
```

本实例运行后会显示 7 个重复的句子，且字体逐行增加，在浏览器中的运行结果如图 7-12 所示。

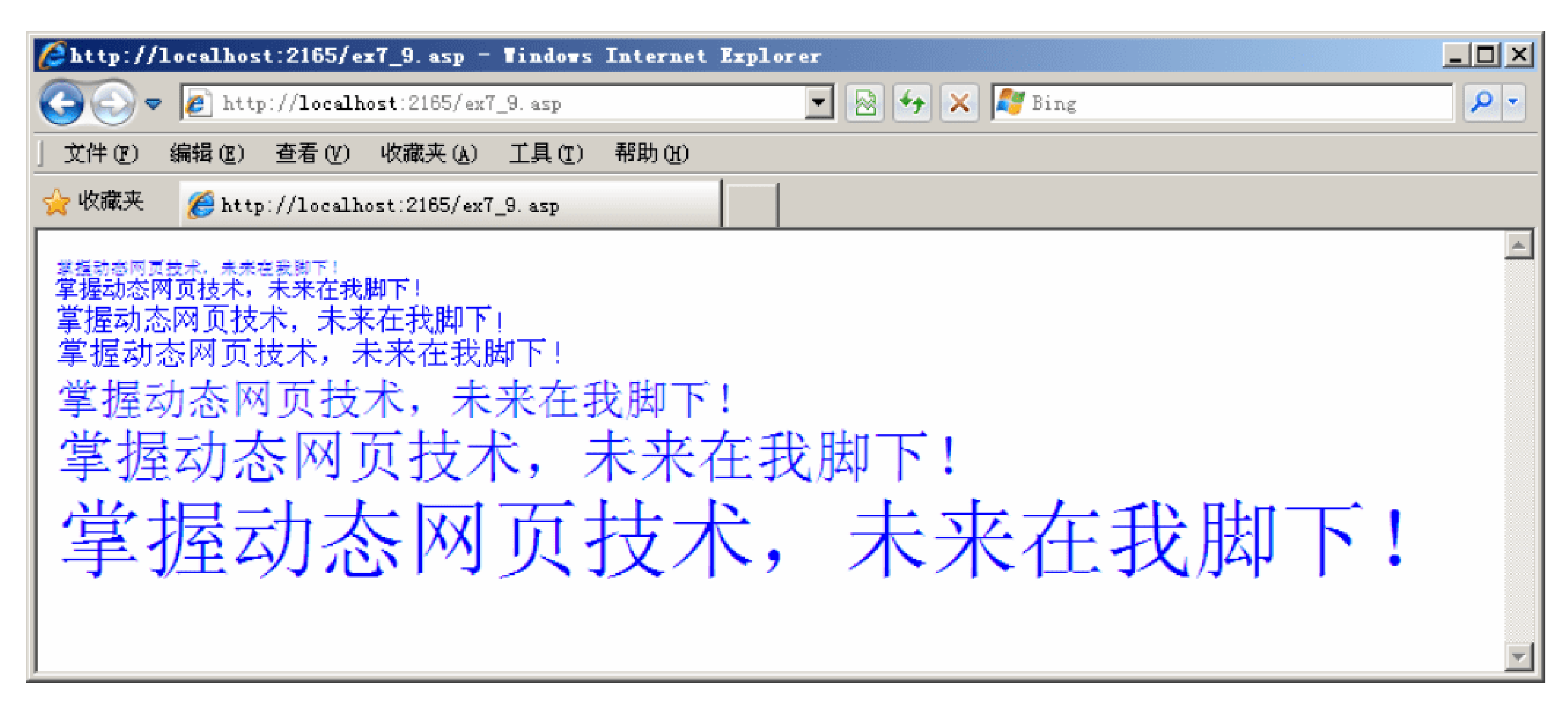


图 7-12　重复循环显示

7.5.4　PHP

1. PHP 是什么

PHP(Personal Hypertext Preprocessor)，即超文本预处理器。1995 年，Rasmus Lerdorf 为了创建他的在线简历而创造了一个“个人主页工具”(Personal Home Page Tools)，这是一种非常简单的语言。1997 年，Zeev Suraski 和 Andi Gutamns 对其加以完善。其后越来越多的人注意到了这种语言并对其提出了各种建议并做了扩展。在许多人的不断完善下以及这种语言本身的源代码的开放特性，使它逐步演变成一种特点丰富的语言，且现在还在不断成长。

PHP 是完全免费的，用户可以从 PHP 官方站点(http://php.net/)自由下载。它在大多数 UNIX 平台、GUN/Linux 和微软 Windows 平台上均可以运行。关于在 Windows 或 Linux 环境下安装 PHP 的方法可以在相关网站上找到。

2. PHP 的版本及其特色

PHP 最新是 PHP 5，PHP 5 相比 PHP 4，是一个飞跃。PHP 5 处理对象部分的内核完全重新开发过，提供更多功能的同时也提高了性能。

其中 PHP 5 三大特色功能为：

- 新的对象模式 (New Object Mode)；
- 异常处理 (Exceptions)；
- 名称空间 (Namespace)。

熟悉.NET、JAVA、C++等面向对象开发的读者会发现 PHP 5 已经完全面向对象化了。

3. PHP 开发相关应用系统的整合

- PHP 开发组合通常是：PHP+Mysql+Zend+IIS/Apache。
- MySQL：是一套优秀的开源数据库系统。PHP 支持各种类型的数据库，但由于 PHP 和 Mysql 都归于开源软件，两者结合在 Web 开发上表现优异。
- Zend 优化器：可以对 PHP 代码加密，保护 PHP 代码的安全性，更重要的是 Zend 优化器可以极大地提高 PHP 程序的运行效率。经过 Zend 优化器优化后的代码比未加密优化的代码运行效率可以提高 3~10 倍。
- IIS/Apache Web 服务器：IIS 是 Microsoft 提供的优秀的 Web 服务器。性能稳定安全，功能强大。Apache 是一个优秀的开源 Web 服务器，在 Linux 上应用广泛。

4. PHP 应用误区

- 误区 1：PHP 只在 Linux+Apache 平台上运行。实际上 PHP 可以在各种流行平台上运行。 Windows/Linux 都是可以支持的，Windows+IIS+PHP 5 的运行性能表现绝对可以和 Linux+Apache+PHP 相同甚至更高，并且安全上更加出色。
- 误区 2：PHP 使用得很少。国外很多网站都是以 PHP 开发的，这与国内的情况形

成鲜明的对比，现在国内 PHP 的应用正慢慢扩大。实际国内很多大型的网络公司都使用了 PHP 进行开发，典型的有百度、淘宝、新浪等。

5. PHP 的优点

(1) 学习过程简单。PHP 的学习过程非常简单。与 Java 和 Perl 不同的是，对于 PHP 只需了解基本的语法和特色，就可以开始编码了，编码过程中遇到的问题可以查阅相关文档来解决。

PHP 的语法与 C、Perl、ASP 或者 JSP 类似。对于那些对上述语言比较熟悉的人来说，PHP 非常简单，PHP 的核心语法通常只需要 30 分钟就可以基本掌握。

(2) 数据库连接。PHP 可以编译成具有与许多数据库相连接的函数。PHP 与 MySQL 是现在绝佳的组合，当然也可以自己编写外围的函数取间接存取数据库。通过这样的途径，在更换不同的数据库产品时，就可以轻松地更改编码以适应变化。PHPLIB 就是最常用的可以提供一般事务需要的一系列基库。

(3) 可扩展性。就像前面说的那样，PHP 已经进入了一个高速发展的时期。对于一个非程序员来说为 PHP 扩展附加功能可能会比较难，但是对于一个 PHP 程序员来说并不困难。

(4) 面向对象编程。PHP 提供了类和对象。基于 Web 的编程工作非常需要面向对象编程能力。PHP 支持构造器、提取类等。

(5) 可伸缩性。传统上网页的交互作用是通过 CGI 来实现的。CGI 程序的伸缩性不很理想，因为它为每一个正在运行的 CGI 程序开一个独立进程。对此问题的解决方法是将经常用来编写 CGI 程序的语言的解释器编译进 Web 服务器(比如 mod_perl，JSP)，内嵌的 PHP 可以具有更高的可伸缩性。

PHP 的开发者们为了更适合 Web 编程，还开发了许多外围的类库，这些库中包含了更易用的层。当然可以利用 PHP 连接包括 Oracle、MS-Access、MySQL 在内的大部分数据库，也可以编写程序下载或者显示 E-mail，甚至完成其他更多的功能。

7.5.5　不同开发技术之间的比较

表 7-1 对 4 种常见的开发方式进行了比较。

表 7-1　不同开发方式的比较

后台界面	CGI	ASP	PHP	JSP/Servlet
操作系统	几乎所有	Win32	几乎所有	几乎所有
服务器	几乎所有	IIS	非常多	非常多
执行效率	慢	快	很快	极快
稳定性	高	中等	高	非常高
开发时间	长	短	短	中等
修改时间	长	短	短	中等
程序语言	不限，几乎所有	VB	PHP	仅支持 Java
网页结合	差	优	优	优
学习门槛	高	低	低	较高

(续表)

后台界面	CGI	ASP	PHP	JSP/Servlet
函数支持	不定	少	多	多
系统安全	佳	低	佳	极佳
使用网站	多	多	超多	目前一般
更新速度	无	慢	快	较慢

JSP 同样是实现动态网页的一个利器。由于脚本语言是 Java，所以继承了 Java 诸多优点。由表 7-1 可知，ASP 与 JSP 基本不在一个档次上，后者要优秀的多，但 ASP.NET 和 Java 却是可以相抗衡的。

1. JSP 与 ASP

无论在运行速度、运行开销、运行平台、扩展性、安全性、函数支持、厂商支持、对 XML 的支持等方面，ASP 都不是 JSP 的对手。COM 组件的复杂性使得这种方式有一定的难度，而 JAVABeans 和 Java 的结合却是天衣无缝的。

2. JSP 与 ASP.NET

以下试从几个不同方面对这两种技术进行分析和比较。

(1) 面向对象。与 Java 一样，ASP.NET 支持 C#等几种面向对象编程语言。C#将成为微软所推出的与 Java 类似的一种语言，与 Java 相比，它可以和 Windows 环境紧密集成，且它的性能更好。

(2) 数据库连接。ASP 另一个亮点是它使用 ADO、ODBC 和 OLE-DB 等事务处理管理器，因此用它开发 Web 数据库应用特别简单。而 ASP.NET 具备了更多的功能，比如 ADO+带来了更强大更快速的功能。JSP 和 JDBC 目前在易用性和性能上同 ASP/ADO 相比已有些落后，当新版本 ASP/ADO+出现后这样的差别可能还会更加明显。

(3) 大型站点应用。ASP.NET 对于企业级的大型站点有更好的支持。事实上，微软在这方面付出了巨大的努力。ASP.NET 考虑到多服务器(multiple servers)的场合，只需要增加服务器就可以在一定范围内增加性能，而且整个.Net 框架已经充分地提供了这个方法。ASP.NET 提供了外部会话状态(External Session State)来对大型站点进行支持。此外，由于请求的各组件相互间经过了充分的优化，所以速度较快。

ASP.NET 现在可以在大型项目方面具有与 JSP 几乎相同的能力。此外，ASP.NET 还有价格方面的优势，因为所有的组件将是服务器操作系统的一部分。对于 JSP，则需要购买昂贵的应用服务器群才能达到同样的目的。

(4) ASP.NET 还提供了更多方面的新特性，包括：

- 内置的对象缓存和页面结果缓存；
- 内置的 XML 支持，可用于 XML 数据集的简单处理；
- 服务器控制提供了更充分的交互式控制。

(5) JSP 的优势。首先，JSP 有一项全新的技术——Servlet(服务器端程序)，这很好地节

约了服务器资源。其次，Java 良好的跨平台特性是 ASP.NET 所不具备的。

3. PHP 与 ASP.NET

提到 Web 开发，很多技术都涉及预处理，即利用特定的标记将代码嵌入到 HTML 页面中，这些标记告诉预处理器，它们包含代码，需要对它们预先作出处理。与 CGI 类似，这些代码都是在服务器端运行的。开放源码的脚本语言 PHP 和 ASP.NET 框架中的语言都属于这种类型。实际上，JSP 和 Perl 也是以这种方式运行的。

与 ASP.NET 相比，PHP 5 仍然存在一些缺点，包括缺少异常和基于事件的错误处理。另一个弱点是 PHP 的函数名是不区分大小写的，虽然这不是一个致命的问题，但这一点可能带来隐藏的错误。另外，PHP 也不是专门设计为一种面向对象的语言。

PHP 的优势在于，它是开放源码的，可以自行修改。此外，PHP 可以与 Apache 很好地协调工作(可以作为一个模块编译，或直接编译成 Apache 二进制文件)，而 Apache 能运行在 Windows、Linux、Solaris 和其他 UNIX 平台上。此外，使用拥有 Apache 的跟踪记录的 Web 服务器意味着安全性能够保持在最高的优先级上。最后，PHP 拥有更小的代码路径，这意味着更少的分析和执行 PHP 页面服务器端代码，这将带来更高效的内存和使用率以及更快的运行速度。

7.6　本章小结

本章主要介绍了用于 Web 开发中的各种动态网页技术，包括 CGI、ASP、ASP.NET、Java Servlet、Java Applet、Servlet、JSP、J2EE、PHP 和 ISAPI/NSAPI 等。读者可以结合实例，进行横向比较，从而了解各种技术的基本特点、所应用的平台、优势以及可能存在的不足之处等。

7.7　思考和练习

1. Java Applet 与前面介绍的多种动态网页技术最大的不同之处是什么?

2. 如果希望在 Linux 操作系统下进行网站的开发和运行，目前有哪些动态网页技术可以选择？试分别比较其优缺点。

3. 针对一个特定的开发需求，采用什么流程可以正确抉择出合适的开发方案?

第8章　Web展望

尼葛洛庞帝(Negroponte)在其《数字化生存》中曾经有一个著名的论述，“网络上的东西将比人要多”。而 P2P 将使得这些“东西”之间的直接交流成为可能，且网络上每个设备都是“活跃”的。今天的互联网已经历了翻天覆地的重大改变。最早它只有基于文本的简单浏览器，供科学家之间交流研究心得；如今，它已经成为商务和信息的中心。在这个过程中，许多新方法和新技术涌现，从早期的图形化浏览器到最近的博客和播客等。现在，互联网已经成为大量应用的首选平台。此时，人们提出了 Web 2.0，它以 Flickr、Craigslist、Linkedin、Tribes、Ryze、Friendster、Del.icio.us 和 43Things.com 等网站为代表，以 Blog、TAG、SNS、RSS 和 wiki 等应用为核心，依据六度分隔、XML 和 Ajax 等新理论和技术实现的互联网的新一代应用模式。

本章旨在让读者掌握 Web 技术发展的最新动态，为今后进一步的学习指明方向。目前，除了本书前面章节中重点介绍的几大技术——HTML、CSS、JavaScript 和动态网页之外，代表 Web 发展方向的技术主要包括 XML、Ajax、客户端开发框架、移动开发等。本章就从这几个方面出发，希望读者能对这些新技术有一定的了解。

本章要点：

- XML 及其相关技术
- Ajax 技术及其开发
- 客户端开发框架
- 移动开发

8.1　Web 的进化路径

人们在享受技术发展所带来的方便的同时，也越来越依赖于技术的进步，但个体掌握、认识和理解技术进步的能力却各不相同。Web 系统是人类迄今为止最伟大的发明之一，也是计算机影响人类最为深远的技术之一。那么，Web 是如何发展起来的呢？

1. 信息共享

虽然人们为信息共享已经奋斗了很多年，但直到 Web 技术的出现并逐步完善的今天，信息共享也还远未令人满意。但比起之前的其他技术，如 FTP 等，自描述性赋予了 Web 系统强大的生命力，使得 Web 成为信息共享的关键设施之一。

2. 信息共建

之前阶段的信息交换主要是单向的，信息是从话语权集团单向传播到受众的。受众几乎没有话语权。Web 的出现逐渐改变了这个现状，造就了意识表达空前活跃的意识空间。

3. 知识传承

计算机是人类的意识外化，其每一次进步，都必然聚合了更多人的智慧。集聚人类智慧为人类共享，是计算机科学技术的内在本质。这里不仅要消灭陷阱病毒，剔除垃圾信息，更要有序化、系统化整个 Web 世界。当知识的累积达到一定数量，以全 Web 资源为基础就自然形成了一座“Web 图书馆”，实现人类自身的“知识传承”。此外，搜索引擎的崛起，为海量知识的检索和快速查找提供了帮助。

4. 知识分配

此阶段之前，人类可以随心所欲地获取各种知识，当然这些知识都是前辈们的贡献。与传统的知识分配方式不同，学生可以不用像传统的一位教师对多位学生进行批量、统一的知识传授的方式来获取知识，而是能制定个性化的学习策略，进行自主学习。比如一个 10 岁的孩子想在 20 岁的时候成为核物理学家，那么在现在的知识体系下，他会怎样学习知识呢？这个问题就是此阶段的核心——知识分配系统能解决的问题。

5. 语用网

技术的发展虽然令人眼花缭乱，但其背后的本质却十分简单。现有的计算机技术都是符合图灵模型的。简单地，图灵机就是机械化、程序化，或者说算术，以数据和算符(算子)为二元的闭合理论体系。图灵机是研究和定义在数据集上的算子规律或法则的数学科学。在网络世界里，这些封闭系统都要联合起来，成为一个整体，即所谓的整个网络成为一台计算机系统。而这台计算机就不再是图灵机了，而是 Petri 网了。早在 20 多年前，Petri 就说过，实现 Petri 网的计算机系统技术叫语用学。由此，语用网才是计算机的技术基础。

6. 云计算和物联网

这些本质上不是单纯的互联网技术或衍生思想。在云计算条件下，以按需、易扩展的方式获取所需要的计算能力或资源，在使用者看来计算能力或资源是可以无限扩展并可以随时获取的。由此形成了虚拟化(Virtualization)、效用计算(Utility Computing)、IaaS(基础设施即服务)、PaaS(平台即服务)、SaaS(软件即服务)等概念。物联网与互联网的初步结合，则构成了一种全新的模式，惠及所有人。而物联网是与互联网等价的物理媒介，其用户端延伸和扩展到了任何物品与物品之间，进行信息交换和通信，也被称为“物物相连的互联网”，它通过智能感知、识别技术与普适计算、广泛应用于网络的融合，构成一个人与物、物与物相连，具备远程管理控制和智能化的网络。云计算和物联网以及将来更多技术的出现，必将构成改变世界的新物理模式，其中的每个人都有调动自己感官的无限权利，用自己的五官去重新发现世界，最终改变整个世界。

8.2 XML 及其相关技术

首先 XML 是一种元标记语言，所谓“元标记”就是开发者可以根据自己的需要定义自己的标签，比如开发者可以定义标签<book>、<name>等，任何满足 XML 命名规则的名称都可以作为标签，这就为不同的应用程序打开了一扇整合之门。另外，由于不同开发者对于各自的不同要求的应用需求可以使用不同结构的 XML 文档，并且可以利用多个不同的 XSLT 从一个已经定义的 XML 文档抽取到需要的数据后，组成不同的显示格式，所以从这个意义上来说，XML 兼具了数据交换和将内容与形式分开的双重作用。

8.2.1 什么是 XML

XML(eXtensible Markup Language)，即可扩展标识语言，在国内很多人将 XML 理解为 HTML 的简单扩展，尽管 XML 同 HTML 关系非常密切，但这实际上是一种误解。

1. XML 与 SGML 和 HTML

SGML、HTML 是 XML 的先驱。如前文所述，SGML(Standard Generalized Markup Language)，即通用标识语言标准，它是国际上定义电子文件结构和内容描述的标准，是一种非常复杂的文档结构，主要用于大量高度结构化数据和其他各种工业领域，能利于分类和索引。同 XML 相比，定义的功能更加强大，但其缺点是不适用于 Web 数据描述，而且 SGML 软件价格非常昂贵。

读者已经比较熟悉 HTML(Hyper Text Markup Language)，即超文本标识语言，它的优点是比较适合 Web 页面的开发。其缺点是标签种类相对较少，只有固定的标签集如<P>、<TABLE>等；缺少 SGML 的柔性和适应性；不能支持特定领域的标记语言，如对数学、化学、音乐等领域的表示支持较少。举个例子来说，开发者很难在网页上表示数学公式、化学分子式和乐谱。

而 XML 则从 Web 运用的角度出发，结合了 SGML 和 HTML 的优点并消除了其缺点。XML 仍然被认为是一种 SGML 语言，但它比 SGML 简单，又能实现 SGML 的大部分功能。图 8-1 形象地表示了三者之间的关系，XML 和 HTML 都是 SGML 的子集，但是 XML 和 HTML 有一部分是重合的，这是因为有一部分 HTML 也可以说是 XML，反之亦然。

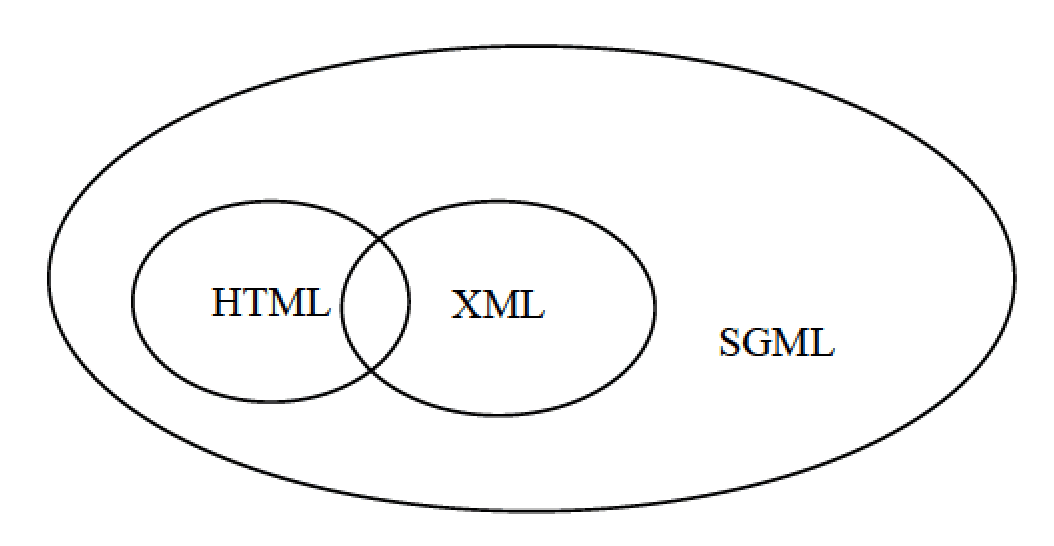


图 8-1 XML 与 SGML 和 HTML 三者间的关系

早在 1996 年的夏天，Sun Microsystem 的 John Bosak 开始开发 W3C SGML 工作组(现在称为 XML 工作组)。他们的目标是创建一种 SGML，使其在 Web 中既能利用 SGML 的

长处，又保留 HTML 的简单性，现在这个目标基本达到了。

2. XML 的发展

在专业领域，出现了不同的 Web 标记语言，比较著名的有 CML——化学标记语言，由 Peter Murray-Rust 开发，同时开发了第一个通用 XML 浏览器 Jumbo。在数学方面，包括 IBM 公司在内都在致力于开发 MathML，在 1997 年 4 月出版了 xll 的第一个版本。1997 年 8 月，Microsoft 公司和 Inso 公司引入 xsl。1998 年 1 月，Microsoft 公司出版发行了 msxsl 程序，它可以利用 XSL 表和 XML 文档创建能被 IE4 识别的 HTML 页面。1998 年 2 月，W3C 发布了 XML1.0 的正式版本。

近几年来，由于网络应用的飞速发展，XML 的发展非常迅猛，出现了 DOM(Document Object Model)、XSLT(XSL Transformation)等新技术，XML 的应用软件也有了飞速的进步。Microsoft、IBM、Breeze 等公司纷纷推出了自己的解析器或开发平台。在 Microsoft、IBM、HP 等大公司的推动下，目前有两个著名的 XML 研究组织，分别是 biztalk.com 和 oasis.org，由他们向 W3C 提出标准的建议。其中 biztalk 是有 Microsoft 牵头组织的，有趣的是 Microsoft 公司同时参加了 oaisis，用 Microsoft 发言人的话来说，就是“一切视 oasis 的发展而定！”。

3. XML 的真实面目

HTML 是一种预定义标签语言，它只认识诸如<HTML>，<P>等已经定义的标签，对于用户自己定义的标签是不认识的。而 XML 是一种语义/结构化语言，它描述了文档的结构和语义。下面的代码是用 HTML 描述一辆车的方法：

```
<html>
  <Title>关于车</Title>
  <Head>车</Head>
  <body>
    <li>品牌：桑塔纳<ul>
    <li>制造商：上汽集团
    <li>容量：1.8
    <li>型号：3000<ul>
  </body>
</html>
```

在 XML 中，同样的数据表示为：

```
<car>
  <brand>桑塔纳</brand>
  <manufactory>上汽集团</manufactory>
  <capacity> 1.8</capacity>
  <model>3000</model>
</car>
```

通过对上面两种方式的对比，可以看出，XML 的文档是具有语义描述能力且结构化的。从低级的角度看，XML 具有一种通用的数据格式。从更高级的层面来看，它是一种自描述语言。

XML 可用于数据交换，这主要是因为 XML 表示的信息是独立于平台的，这里的平台既可以理解为不同的应用程序也可以理解为不同的操作系统；它描述了一种规范，利用它

可以完成 Microsoft 的 Word 文档与 Adobe 的 AcrobatPDF 格式文件的相互交换信息，也可以在异构的数据库之间交换信息。

对于大型复杂的文档，XML 是一种理想语言，它不仅允许指定文档中的词汇，还允许指定元素之间的关系。比如可以规定一个 author 元素必须有一个 name 子元素。可以规定企业的业务必须包括什么子业务。

8.2.2 XML 的文档格式

1. 元素

XML 文档内容的基本单元为元素，它的语法如下：

```
<标签>文本内容</标签>
```

元素由起始标签、元素内容和结束标签组成。用户把要描述的数据对象放在起始标签和结束标签之间。例如：

```
<姓名>张三</姓名>
```

无论文本内容有多长或者多么复杂，XML 元素中还可以再嵌套别的元素，这样使相关信息构成等级的结构。下面的例子中，在<中华人民共和国公民>的元素中包括了所有公民的信息，每位公民都由<身份证号>、<姓名>和<籍贯>三个元素来描述，在这个层次结构中，<中华人民共和国公民>元素中又嵌套了<公民>，而<公民>元素中嵌套了<身份证号>、<姓名>和<籍贯>元素。

【实例 8-1】XML 文档实例

程序代码如 ex8_1.xml 所示。

ex8_1.xml

```
<?xml version="1.0" encoding="gb2312" standalone="yes"?>
<中华人民共和国公民>
  <公民>
    <身份证号>01085041</身份证号>
  <姓名>老吕</姓名>
  <籍贯>北京</籍贯>
  </公民>
  <公民>
    <身份证号>02085063</身份证号>
    <姓名>老邱</姓名>
    <籍贯>上海</籍贯>
  </公民>
</中华人民共和国公民>
```

运行后会浏览器上的显示如图 8-2 所示，显示的文字具有不同的颜色，且有些元素前面带有“—”，这意味该元素是可以展开和折叠的，界面上的缩进代表了元素的从属关系。

除了元素，XML 文档中能出现的有效对象包括处理指令、注释、根元素、子元素和属性，下面对此做个简单的介绍。

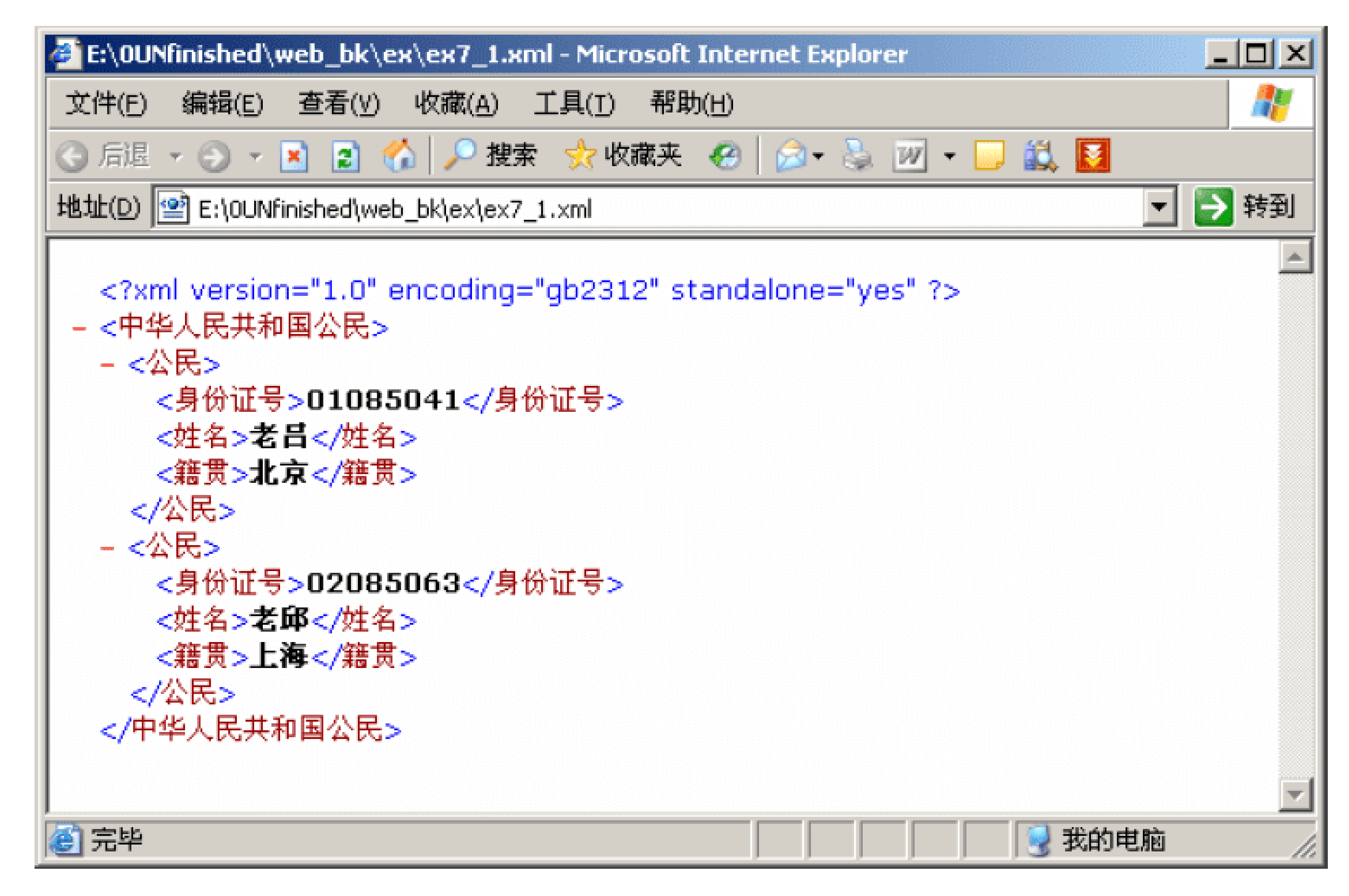


图 8-2　XML 文档在浏览器中的显示

2. 处理指令

处理指令给 XML 解析器提供信息，使其能够正确解释文档内容，它的起始标识是“<?”，结束标识是“?>”。常见的 XML 声明就是一个处理指令，例如：

```
<?xml version="1.0"?>
```

处理指令还有其他的用途，比如定义文档的编码方式是 GB2312 码还是 Unicode 编码方式，或是把一个样式表文件应用到 XML 文档上用以显示。

3. 注释

注释是 XML 文件中用作解释的字符数据，XML 处理器不对它们进行任何处理。注释是用“<!--”和“ -->”引起来的，可以出现在 XML 元素间的任何地方，但是不可以嵌套，例如：

```
<!--这是一个注释-->
```

4. 根元素和子元素

如果一个元素从文件头的序言部分之后开始一直到文件尾，包含了文件中所有的数据信息，可被称为根元素。一个 XML 文档中有且仅有一个根元素，其他所有的元素都是它的子元素，【实例 8-1】中，<中华人民共和国公民>就是根元素。

XML 元素是可以嵌套的，那么被嵌套在内的元素称为子元素。在前面的例子中，<公民>就是<中华人民共和国公民>的子元素。

5. 属性

属性给元素提供进一步的说明信息，它必须出现在起始标签中。属性以名称/取值对出现，属性名不能重复，名称与取值之间用等号“=”分隔，并用引号把取值引起来。例如：

```
<salary currency="US$"> 25000 </salary>
```

此外的属性说明了薪水的货币单位是美元。

6. “格式良好”与“有效”的 XML 文档

一个“格式良好”的 XML 文档除了要满足根元素唯一的特性之外，还包括如下内容。

- 起始标签和结束标签应当匹配：结束标签是必不可少的。
- 大小写应一致：XML 对字母的大小写是敏感的，<name>和<Name>是完全不同的两个标签，所以结束标签在匹配时一定要注意大小写一致。
- 元素应当正确嵌套：子元素应当完全包括在父辈元素中。
- 属性必须包括在引号中。
- 元素中的属性是不允许重复的。

而一个“有效”的 XML 文档是指一个 XML 文档应当遵守 DTD 文件或是 Schema 的规定，“有效的”XML 文档肯定是“格式良好的”。

8.2.3 XML 相关技术介绍

1. DTD

XML 文档可由 DTD 和 XML 文档组成。所谓 DTD(Document Type Definition)，即文档类型定义，简单的说就是一组标记符的语法规则，它表明 XML 文本是怎样组织的。比如 DTD 可以表示一个<book>必须有一个子标签<author>，可以有或者没有子标签<pages>等。当然一个简单的 XML 文本是可以没有 DTD 的。

2. XML Schema

XML 语言必须有其严格的规范，以适应广泛的应用。XML 文档必须是“格式良好”的，这一点是很容易被验证的。与此同时，在特定的应用中，数据本身在含义、数据类型、数据关联上的“有效”是较为困难的。

以前，这种限制只有一种定义方式，即上面提到的 DTD 使用了一种特殊的规范来定义在各种文件中使用 XML 标签的规范。但是，有许多常用的限制不能用 DTD 来表述。

此外，尽管 DTD 给标签的使用增加了限制，但是对于 XML 的自动处理却还需要更加严格、更加全面的工具。比如 DTD 不能保证一个标签的某个属性的值必须不为负值，于是出现了 XML Schema。Schema 相对于 DTD 的明显好处是 XML Schema 文档本身也是符合规范的 XML 文档，而不是像 DTD 那样使用特殊格式。这就大大方便了用户和开发者，因为这样就可以使用相同的工具来处理 XML Schema 和其他 XML 文档，而不必专门为 Schema 使用特殊工具。DTD 对用户来说是额外的；而 Schema 却简单易懂。

最初 XML Schema 由 Microsoft 提出，W3C 的专家们经过充分讨论和论证，在 1999 年的 2 月，发布了一个需求定义，说明 Schema 必须符合的要求。同年 5 月，W3C 完成并发布了 Schema 的定义。目前，IE 5 之后版本中的 XML 解析器就能够根据文档类型定义(DTD)或 XML Schema 来解析和验证 XML 文档。

3. CSS、XSL 和 XSLT

通过前面的介绍可以知道，XML 可以定义信息的内容，却没有说明该信息如何展示，这实际上是 XML 的长处，它把内容和形式分离了。因为这个原因，一个内容可以有不同的展现形式，相信随着 XML 应用的提高，那种“建议使用 1024×768 分辨率”的话语会逐渐消失。而 XML 内容的展示就是通过 XSL(XML Style Language)或 CSS(Cascading Style Sheets)来实现的。CSS 与 XSL 的区别如表 8-1 所示。

表 8-1　CSS 与 XSL 的区别

比 较 类 型	CSS	XSL
适用在 HTML	可以	不行
适用在 XHTML	可以	可以
适用在 XML	可以	可以
使用的语法	CSS 样式语法	XML 语法
是否是转换语言	不是	是

【实例 8-2】XML+CSS 文档实例

程序代码如 ex8_2.xml 及 ex8_2.css 所示。

ex8_2.xml

```
<?xml version="1.0" encoding="gb2312" standalone="yes" ?>
<?xml-stylesheet href="ex8_2.css" type="text/css"?>
<DOG_data>
   <unit1><DOG>
      <fontstyle1>姓名：</fontstyle1>
      <Nickname>波波</Nickname><br/>
      <fontstyle1>主人：</fontstyle1>
      <Breeder>盖瑞</Breeder><br/>
      <fontstyle1>生日：</fontstyle1>
      <Birthday>10/17</Birthday><br/>
      <fontstyle1>年龄：</fontstyle1>
      <HowOld>3 岁</HowOld><br/>
      <fontstyle1>品种：</fontstyle1>
      <Breed>Husky</Breed><br/>
      <textblock>
         <fontstyle1>特性：</fontstyle1>
         <character>好吃懒做爱睡觉，很臭屁，看到别的狗就扑过去。至今未尝败绩，希望能有一只狗可打败它。</character>
      </textblock>
   </DOG></unit1>
   <unit2><DOG>
      <fontstyle2>姓名：</fontstyle2>
      <Nickname>汪汪</Nickname><br/>
      <fontstyle2>主人：</fontstyle2>
      <Breeder>小王</Breeder><br/>
      <fontstyle2>生日：</fontstyle2>
      <Birthday>6/06</Birthday><br/>
```

```
        <fontstyle2>年龄：</fontstyle2>
        <HowOld>3 岁</HowOld><br/>
        <fontstyle2>品种：</fontstyle2>
        <Breed>土狗</Breed><br/>
        <fontstyle2>特性：</fontstyle2>
        <character>逊狗一只，土狗是指很土的狗。</character>
    </DOG></unit2>
</DOG_data>
```

ex8_2.css

```
unit1
{ display:block;
    text-align:left;
    color:black;
    background:#FFE1FF;
    padding:2cm;
    width:15cm;
}
unit2
{ display:block;
    color:black;
    background:#E0EEE0;
    padding:1cm;
    width:15cm;
}
fontstyle1
{ font:italic12ptextra-bold;
    color:red;
}
fontstyle2
{ font:italic12ptextra-bold;
    color:green;
}
br
{ display:block;
}
textblock
{ border:black2pxsolid;
    float:right;
    padding:10px;
    width:5cm;
}
```

本实例中处理指令<?xml-stylesheet href="ex8_2.css" type="text/css"?>将外部定义的 CSS 引入。其中 CSS 在第 5 章已做过介绍。当浏览器打开 ex8_2.xml 的同时也打开了 ex8_2.css，按照设定的几种不同样式来显示 XML 文件中的数据，最终在浏览器中的显示效果如图 8-3 所示。

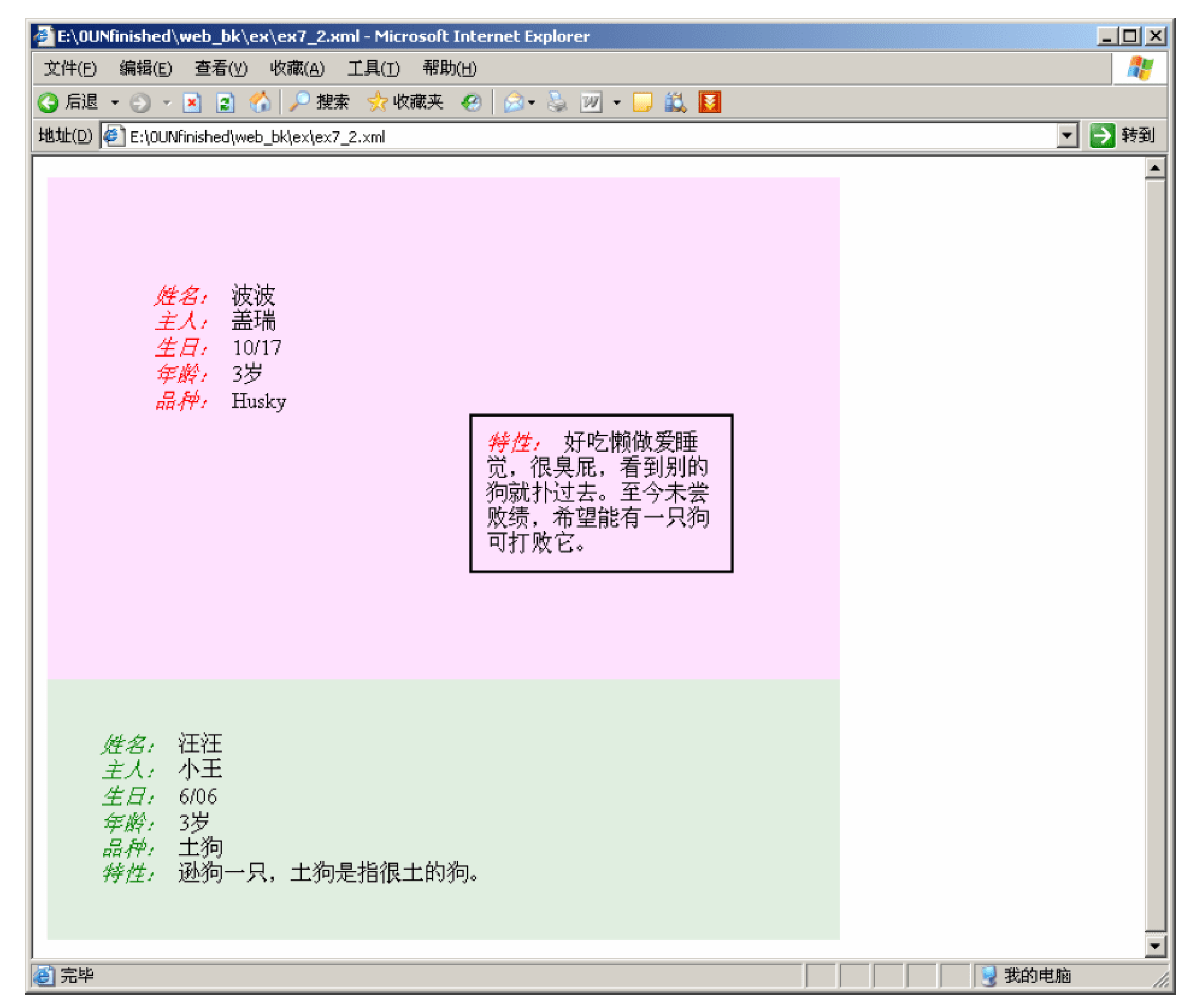


图 8-3　XML+CSS 文档实例

【实例 8-3】XML+XSL 文档实例

程序代码如 ex8_3.xml 及 ex8_3.xsl 所示。

ex8_3.xml

```
<?xml version="1.0" encoding="gb2312" ?>
<?xml-stylesheet href="ex8_3.xsl" type="text/xsl"?>
<Cars>
  <car>
    <BrandName>甲车</BrandName>
    <Dealership>A 厂</Dealership>
    <Price>500</Price>
    <Amount>3</Amount>
  </car>
  <car>
    <BrandName>乙车</BrandName>
    <Dealership>B 厂</Dealership>
    <Price>200</Price>
    <Amount>7</Amount>
  </car>
  <car>
    <BrandName>丙车</BrandName>
    <Dealership>A 厂</Dealership>
    <Price>300</Price>
    <Amount>4</Amount>
  </car>
</Cars>
```

ex8_3.xsl

```
<?xml version="1.0" encoding="gb2312"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
<xsl:template match="/">
<HTML>
<HEAD></HEAD>
<BODY>
  <TABLE BORDER="1">
  <TR>
    <TH>车名</TH>
    <TH>经销商</TH>
    <TH>售价</TH>
    <TH>数量</TH>
  </TR>
  <xsl:for-each select="Cars/car">
  <TR>
    <xsl:apply-templates />
  </TR>
  </xsl:for-each>
  </TABLE>
</BODY>
</HTML>
</xsl:template>

<!-- 车名样板 -->
<xsl:template match="BrandName">
  <TD STYLE="COLOR:green"><xsl:value-of /></TD>
</xsl:template>

<!-- 其他样板 -->
<xsl:template match="Dealership">
  <TD><xsl:value-of /></TD>
</xsl:template>

<xsl:template match="Price">
  <TD><xsl:value-of /></TD>
</xsl:template>

<xsl:template match="Amount">
  <TD><xsl:value-of /></TD>
</xsl:template>

</xsl:stylesheet>
```

本实例中处理指令<?xml-stylesheet href="ex8_3.xsl" type="text/xsl"?>将外部定义的 XSL 引入。当浏览器打开 ex8_3.xml 的同时也打开了 ex8_3.xsl，按照 xsl 中设定的样式来显示 XML 文件中的数据，由于这里设置了一个表格，因此，最终在浏览器中显示了一个表格，效果如图 8-4 所示。

由于 XSL 文档的转换规则较为复杂，在此就不详细介绍了，有兴趣的读者可以参考专

门的书籍来了解。

XSLT 即 XML Stylesheet Language Transformation。XSLT 是一种用来进行 XML 文档间相互转化的语言。简单的说，不同的开发者对于各自的应用会使用不同的XML文档，XSLT 可以从一个已经定义的 XML 文档抽取需要的数据，转换为不同的形式，可以是 XML、HTML 和各种不同的 SCRIPT。而 XSLT 需要有解析器，才能正确显示 XML 文件，而常用的解析器就是微软的 MSXML。



图 8-4　XML+XSL 文档实例

注意：

在微软的 IE5.5 以上版本内嵌了 MSXML 3 解析器，可以用 IE 5.5 以上版本打开某个.xml 文件，就可以看到转换后的结果了。但是如果只看到没有经过转换的 XML 结构树，这就说明您的浏览器没有安装 MSXML。

删除和恢复 MSXML 的 DOS 命令分别为：

```
resvr32/u msxml3.dll
regsvr32 msxml3.dll
```

4. DOM

DOM 即 Document Object Model，它把 XML 文档的内容实现为一个对象模型，简单的说就是应用程序如何访问 XML 文档，W3C 的 DOM Level 1 定义了如何实现属性、方法、事件等。

面向对象的思想方法已经非常流行了，在编程语言(如 Java、JSP 等)中，广泛使用了面向对象的编程思想。在 XML 中，就是要将网页也作为一个对象来操作和控制，用户可以建立自己的对象和模板。DOM 就是一种详细描述 HTML/XML 文档对象规则的 API。它规定了 HTML/XML 文档对象的命名协定、程序模型、沟通规则等。在 XML 文档中，可以将每一个标识元素看作一个对象——它有自己的名称和属性。XML 创建了标识，而 DOM 的作用就是告诉 JavaScript 如何在浏览器窗口中操作和显示这些标识。

对于 XML 应用开发者而言，DOM 就是一个对象化的 XML 数据接口，一个与语言无关、与平台无关的标准接口规范。它定义了 HTML 文档和 XML 文档的逻辑结构，给出了一种访问和处理 HTML 文档和 XML 文档的方法。利用 DOM，程序开发人员可以动态地创建文档，遍历文档结构，添加、修改、删除文档内容，改变文档的显示方式等。可以这样说，文档代表的是数据，而 DOM 则代表了如何去处理这些数据。无论是在浏览器里还是在浏览器外，抑或是在服务器上还是在客户端，只要有用到 XML 的地方，就会碰到对 DOM 的应用。

作为 W3C 的标准接口规范，目前，DOM 由三部分组成，包括核心(core)、HTML 和 XML。核心部分是结构化文档比较底层对象的集合，这一部分所定义的对象已经完全可以

表达出任何 HTML 和 XML 文档中的数据了。HTML 接口和 XML 接口两部分则是专为操作具体的 HTML 文档和 XML 文档所提供的高级接口，以便对这两类文件的操作更加方便。

XML DOM包含四个主要对象：XMLDOMDocument、XMLDOMNode、XMLDOMNodeList和XMLDOMNamedNodeMap。

5. Xpointer 和 Xlinks

类似于HTML中的Hyper Link，Xpointer和Xlinks用于链接其他XML文档和其他XML文档中的某个部分，其中 Xpointer 相当于 HTML 中用于定位 HTML 文档子内容的锚。不过其链接水平更强大。比如，在 bookstore 中，可以定位到有一个作者叫“吴伟敏”且书中有“XML”的那本书，在 HTML 中，这是不可能实现的。

当然，XML 的发展促使了许多新技术的出现，其他类似的还有 RDF、Xfrom、RSS 等，其中的大部分 W3C 只是给出了建议，还没有形成正式的标准，有些内容甚至还处于探讨阶段。

6. XML 框架

所谓框架即 Framework。XML 是一个通用的标准。它不属于个人，认证它的也不是一家公司，而是 W3C。那么为什么那么多的大公司纷纷趋之如鹜呢？各家公司互相竞争的是它的 Framework 和 Schema。XML Framework 是驾驭 XML 文件的结构，是一种高层次的结构控制。利用 XML Framework，可以把商业逻辑(business logic)分离出来，实现数据与计算的分离。

目前著名的框架有Microsoft的Biztalk以及联合国(UN/CEFACT)和OASIS联合于1999年底推出的 EBXML 动议。相信在不久的将来会有许多的框架，其中的一个问题就是在 W3C 中关于 XML 的很多内容还不成熟时，就推出框架是一种冒险。

8.2.4　XML 的开发工具

为了帮助读者进一步了解和开发 XML，这里对常用的工具进行简要介绍。

- Notepad：最直接、最简单的文本编辑工具，在大多时候操作系统中都有。
- Microsoft XML Notepad：微软专门为设计 XML 文档而提供的编辑软件，可以借助它验证 XML 文档的有效性。目前最新的版本为 2007 版。
- Microsoft XML Tree Viewer：利用这个软件可以把 XML 文档的内容用树的结构形式显示出来。
- Microsoft XML Validator：该软件可以检查 XML 文档是否是“格式良好”及其“有效”性，并对错误发出警告。
- Microsoft XSL Debugger：样式单文件的复杂性使开发人员在编写时容易出现错误，这个软件就是帮助用户调试样式单文件的，把复杂枯燥的调试过程用可视化界面显示出来。
- Xray：一种具有实时错误检查的 XML 编辑器。它根据 DTD 或者 XML Schema，

允许用户创建格式良好的 XML 文档或验证文档的有效性，并且支持多文档编辑，是一款免费软件。

- XMLWriter：一款适合于专业 XML 开发者及初学者的 XML 编辑工具，支持 XML、XSL、DTD/Schema、CSS、HTML 等文本格式的文件编辑和调试。可使用 CSS、XSL 等语言在一个集成的预览窗口中格式化 XML 文件。XML Writer 有一个直观性、个性化的用户界面让使用者编辑，还具有书签功能，可以自动查找并替代。其他的功能还有 XML 联机帮助、插件管理、即时色彩编码转换、树形结构查看、批量及命令行处理、可扩充的能力及更多功能。
- XMLSpy(XML 编辑器)：提供三种 XML 文档视图，分别是结构显示和编辑、原码视图和支持 CSS 和 XSL 的预览。使用它能够设计、编辑和调试含有 XML、XML Schema、XSL/XSLT，SOAP，WSDL 和网络服务技术的企业级的应用程序。其中包含了 XSLT 调试程序、WSDL 编辑、HTML-to-XML 转换、XML 模式驱动的代码生成和 Tamino 集成。它是符合行业标准的 XML 开发环境，专门用于设计、编辑和调试企业级的应用程序，是 J2EE、.NET 和数据库开发人员不可缺少的高性能的开发工具。
- Sonic Stylus Studio：为 XML 开发者提供了更大的生产力，如领先的 XSLT 编辑和 debug 工具、先进的 Xquery 编辑器和 debug 工具，图像化 XML Schema 设计工具、独特的 XML-to-XML mapper、所见即所得的 XML-to-HTML 设计工具，以及支持 Sense:X 自动完成功能。Sonic Stylus Studio 进一步支持 Xquery (2003 年的 W3C 规格)、Web Services Call Composer、XSLT 及 Xquery 档案管理，并支持多种 XML schema 确认(validation)引擎。

8.2.5　XML 的使用前景

不管怎样，Web 的应用将随着 XML 的发展而更加精彩。

1. 商务的自动化处理

XML 的丰富置标完全可以描述不同类型的单据，例如信用证、保险单、索赔单以及各种发票等。结构化的 XML 文档发送至 Web 的数据可以被加密，并且很容易附加上数字签名。因此，XML 有希望推动 EDI(Electronic Data Interchange)技术在电子商务领域的大规模应用。有兴趣的读者可以访问网站 http://www.xmledi.org。

2. 信息发布

信息发布在企业的竞争发展中起着重要作用。服务器只需发出一份 XML 文件，客户就可以根据自己的需求选择和制作不同的应用程序来处理数据。加上 XSL(eXtensible Stylesheet Language)的帮助，使广泛的、通用的分布式计算成为可能。

3. 智能化的 Web 应用程序和数据集成

XML 能够更准确地表达信息的真实内容，其严格的语法降低了应用程序的负担，也

使智能工具的开发更为便捷。来自不同应用程序的数据也能够转化到 XML 这个统一的框架中，进行交互、转化和进一步的加工。

XML 的优点备受瞩目，它的发展方兴未艾，未来的 Web 将是 XML 的 Web！

8.2.6 JSON(JavaScript Object Natation)

虽然 XML 是进行数据交换的标准方式，但通常它不是最好的方式，JSON 为 Web 应用开发者提供了另一种数据交换格式。尽管 XML 可以把结构和元数据添加到数据上，但是它使用了一种相当烦琐的方式。XML 还有一种相对复杂的语法，因而需要一种分析器对之进行专门分析。在 JavaScript 中，XML 必须被分析成一棵 DOM 树。且一旦构建了这棵 DOM 树，还必须在其中添加导航以便创建相应的 JavaScript 对象或者在其他方式客户端 Web 应用程序中使用 XML 数据。同 XML 或 HTML 片段相比，JSON 提供了更好的简单性和灵活性。与 XML 不同的是：JSON 只能用来传输数据，而不能用作文档格式。它是一种轻量级的数据交换格式，非常适合于服务器与 JavaScript 的交互。

尽管 XML 拥有跨平台、跨语言的优势，但除了应用于 Web Services，否则在一般的 Web 应用中，开发者经常为 XML 的解析颇费脑筋，无论是服务器端生成或处理 XML，还是客户端用 JavaScript 解析 XML，常常会需要较为复杂的代码，这无形中降低了开发的效率。实际上，对于大多数 Web 应用来说，是不需要复杂的 XML 来传输数据的，XML 的扩展性在这些项目中不完全具有优势，许多 AJAX 应用甚至直接返回 HTML 片段来构建动态 Web 页面。和返回 XML 并解析它们相比，返回 HTML 片段大大降低了系统的复杂性，但同时也丧失了一部分灵活性。

与 XML 一样，JSON 也是基于纯文本的数据格式。由于 JSON 天生是为 JavaScript 准备的，因此，JSON 的数据格式非常简单，因此可以用 JSON 传输一个简单的 String、Number、Boolean，也可以传输一个数组，或者一个复杂的 Object 对象。

总的来说，JSON 具有：轻量级的数据交换格式、读/写更加容易、易于服务器的解析和生成、能够通过 JavaScript 中 eval()函数解析 JSON。JSON 支持包括：ActionScript、C、C#、ColdFusion、Java、JavaScript、ML、Objective CAML、Perl、PHP、Python、Rebol、Ruby 和 Lua 等多种语言的优点。可以说 XML 比较适合于标记文档，而 JSON 却更适合于进行数据交换处理。

8.3 Ajax 技术

8.3.1 Ajax 的现状

目前 Ajax 已经成为 Web 应用的主流开发技术，大量的业界巨头已经采纳并且在大力推动这个技术的发展。近期的动态包括以下方面。

(1) IBM、Oracle、Yahoo!、BEA、RedHat、Novell 等业界领先的公司启动了 Open Ajax

项目。致力于为 Ajax 开发创建先进强大的开发工具。IBM 在 2 月底已经发布了 Open Ajax 项目的 Ajax Toolkit Framework(ATF)1.0，它是一个基于 Eclipse IDE 的 Ajax 开发工具。

(2) 微软开发了自己的 Ajax 框架 Altas，不过主要是和自己服务器端的 ASP.NET 框架配合工作。今后的 IE 中将拥有 Ajax 的所有内容——DHTML、JScript 和 XmlHttp。2007 年 1 月 25 日，微软发布了 ASP.NET Ajax 1.0(Atlas)的最终版。

(3) Sun 虽然行动迟缓，但是也将 Ajax 技术列入了 J2EE 的 blueprint(蓝图)中，作为 J2EE 技术的有益补充。

除了上述这些公司之外，Google 公司不可不提，因为正是他们率先采用 Ajax 技术创建了一大堆非常出色的应用，才将 Ajax 技术展现在公众面前。Google 建立的 Ajax 应用包括 Google Maps、GMail、Google Suggest 等，其中被公认为最优秀的 Ajax 应用是 Google Maps。由于完全基于 Ajax 技术来构造 Google Maps 的界面，故 Google Maps 提供了远远超越其竞争对手的地图服务的交互体验。如果说 Google 后台的地图技术并不存在巨大优势的话，那么 Ajax 技术和优秀的交互设计则成为他们压倒竞争对手的关键武器。最终使得 Google Maps 脱颖而出，获得了广大用户的青睐。

如果对比微软前后的两个地图服务就可以看出差别。微软所提供的旧的地图服务网址为 http://terraserver.microsoft.com，这是传统 Web 应用的代表，性能较差且易用性不好。而微软新的地图服务网址为 http://local.live.com/，包括该网站上很多其他服务都基于 Ajax 技术，因此获得了极好的可用性。

Ajax 的典型应用除了 Google Maps、微软的 Windows Live 外，还包括 Yahoo！的 Flickr、国内新浪的 blog 等。Web 应用程序将受到更多开发者和用户的青睐，作为领路人的 Google 已经使用了一系列基于 Web 的产品，它们甚至颠覆了传统的网页概念，用户甚至不敢相信基于浏览器的程序竟能实现如此强大的功能。

8.3.2　Ajax 是什么

Ajax(Asynchronous JavaScript and XML)，即异步 JavaScript 和 XML，是指一种创建交互式网页应用的网页开发技术。它并不是一门新的语言或技术，它实际上是几项技术按一定的方式组合在一起共同协作来发挥各自的作用，它包括的内容如下。

- 利用 XHTML 和 CSS 实现标准化的呈现；
- 借助 DOM 实现动态显示和交互；
- 使用 XML 和 XSLT 进行数据的交换与处理；
- 采用 XMLHttpRequest 进行异步数据读取；
- 通过 JavaScript 绑定和处理所有数据。

Ajax 的工作原理相当于在用户和服务器之间加了多个中间层，使用户操作与服务器响应异步化。这样将以前的一些服务器负担的工作转嫁到客户端，利于客户端闲置的处理能力来处理，减轻服务器和带宽的负担，从而达到节约 ISP 的空间及带宽租用成本的目的。

类似于 DHTML，Ajax 不是指一种单一的技术，而是有机地利用了一系列相关的技术。事实上，一些基于 Ajax 的“派生/合成”式(derivative/composite)的技术正在出现，如

“AFLAX”。

Ajax 的应用必须使用支持以上技术的 Web 浏览器作为运行平台。这些浏览器目前包括 Mozilla、Firefox、Internet Explorer、Opera、Konqueror 和 Safari。此外，随着浏览器的发展，更多的技术还会被添加进 Ajax 的技术体系之中。例如，目前 Firefox 浏览器的新版本已经可以直接支持矢量图形格式 SVG。Firefox 已经可以支持 JavaScript 2.0(对应 ECMAScript 4.0 规范)中的 E4X(JavaScript 的 XML 扩展)。Firefox、Opera、和 Safari 浏览器还可以支持 Canvas(也是 Web Applications1.0 规范的一部分)，网络上已经有人开发出了使用 Canvas 技术制作的 3D 射击游戏的演示版。

该技术在 1998 年前后得到了应用。允许客户端脚本发送 HTTP 请求(XMLHTTP)的第一个组件由 Outlook Web Access 小组写成。该组件原属于 Microsoft Exchange Server，并且迅速成为 Internet Explorer 4.0 的一部分。部分观察家认为，Outlook Web Access 是第一个应用了 Ajax 技术的成功的商业应用程序，并成为包括 Oddpost 的网络邮件产品在内的许多产品的领头羊。

直到 2005 年初，所出现的许多事件才使 Ajax 被大众所接受。Google 在它著名的交互应用程序中使用了异步通信，如 Google 讨论组、Google 地图、Google 搜索建议、Gmail 等。Ajax 这个词由 *Ajax: A New Approach to Web Applications* 一文所创，该文的迅速流传提高了人们使用该项技术的意识。另外，对 Mozilla/Gecko 等所提供的支持使得该技术走向成熟，变得更为易用。

Ajax 技术有两个推动力：Web 标准的成熟和软件交互设计及可用性理论的成熟。在软件的可用性方面，除了一些通用的软件可用性和交互设计理论之外(这方面的经典著作包括《面向使用的软件设计》、《About Face 2.0》中文版等)，Web 应用的可用性(Web usability)也是国外非常热门的一个研究领域，其主要侧重于研究如何提高 Web 应用的可用性。美国在这个领域有着非常深入的研究，并且对于一些公共机构网站的可用性还有相关的法律条款来约束(Section508，508 条款，于 2001 年 6 月 21 日成为美国的法律，直接影响了联邦部门和一些代理机构，还有为他们服务的网页设计师。这条法律也适用于政府投资项目和任何采用了该法律的州)。对于这些网站，如果无法达到条款上的一些可用性要求，网站经营者就违法了。如果是开发公司无法达到这些要求，就别指望从联邦政府手中拿到这些项目。

为了对如何提高 Web 应用的可用性做出指导，W3C 在 20 世纪 90 年代建立了 Web Accessibility Initiative(WAI)，致力于为网站创建者提供实现可访问性(与可用性同义)的方法和策略(http://www.w3.org/WAI/GL/)，Web 可用性方面的经典著作包括《网站重构》。

综上所述，可以认为 Ajax 就是 Web 标准和 Web 应用的可用性理论的集大成者。它极大地改善了 Web 应用的可用性和用户的交互体验，最终得到了用户和市场的广泛任可。所以可以说，Ajax 就是用户和市场必然的选择。

8.3.3　与传统的 Web 应用比较

XMLHttpRequest 的出现为 Web 开发提供了一种全新的可能性，甚至完全改变了人们

对于 Web 应用由什么来组成的看法。在这个技术出现之前，由于技术上的限制，人们认为 Web 应用就是由一系列连续切换的页面组成的。因此整个 Web 应用被划分成了大量的页面，其中大部分是一些很小的页面。用户大部分的交互都需要切换并刷新整个页面，而在这个过程中(下一个页面完全显示出来之前)，用户只能等待，什么都做不了。然而 XMLHttpRequest 技术的出现使得开发者可以打破这种笨拙的开发模式，以一种全新的方式来做 Web 开发，为用户提供更好的交互体验。

对于传统 Web 应用与 Ajax 应用在处理用户交互的方式上不同之处的比较如图 8-5 所示。传统的 Web 应用允许用户填写表单(form)，当提交表单时就向 Web 服务器发送一个请求。服务器接收并处理传来的表单，然后返回一个新的网页。这种做法浪费了许多带宽，因为在前后两个页面中的大部分 HTML 代码往往是相同的。由于每次应用的交互都需要向服务器发送请求，应用的响应时间就依赖于服务器的响应时间。这导致了用户界面的响应比本地应用要慢得多。

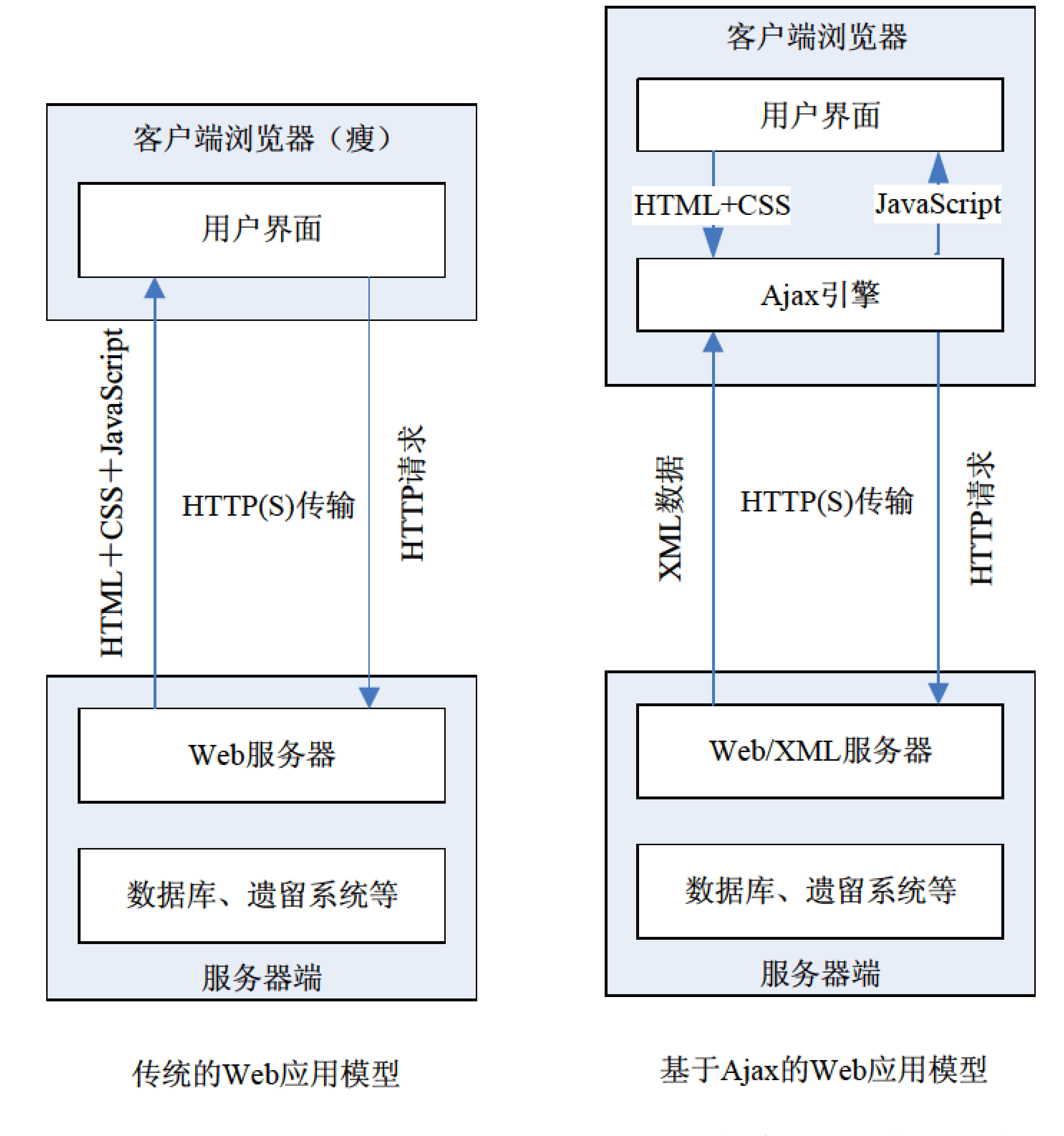


图 8-5　传统 Web 与 Ajax——处理用户交互方式的比较

与此不同，Ajax 应用可以仅向服务器发送并取回必需的数据，它使用 SOAP 或其他一些基于 XML 的 Web Service 接口，并在客户端采用 JavaScript 处理来自服务器的响应。因为在服务器和浏览器之间交换的数据大量减少，结果就能得到响应更快的应用。同时很多的处理工作可以在发出请求的客户端机器上完成，所以 Web 服务器的处理时间也减少了。因此 Ajax 应用与传统的 Web 应用的区别主要表现在 3 个方面：

- 不刷新整个页面，在页面内与服务器通信。
- 使用异步方式与服务器通信，不需要打断用户的操作，具有更加迅速的响应能力。
- 应用仅由少量页面组成。大部分交互在页面之内完成，不需要切换整个页面。

比较图 8-6 与图 8-7，可以发现，Ajax 引擎在这里实际上起到了一个中间层的作用。

由于很多情况下无须等待服务器端的响应，因此减少了网络传输和服务器端处理的时间，正是它使得客户端响应速度加快了。

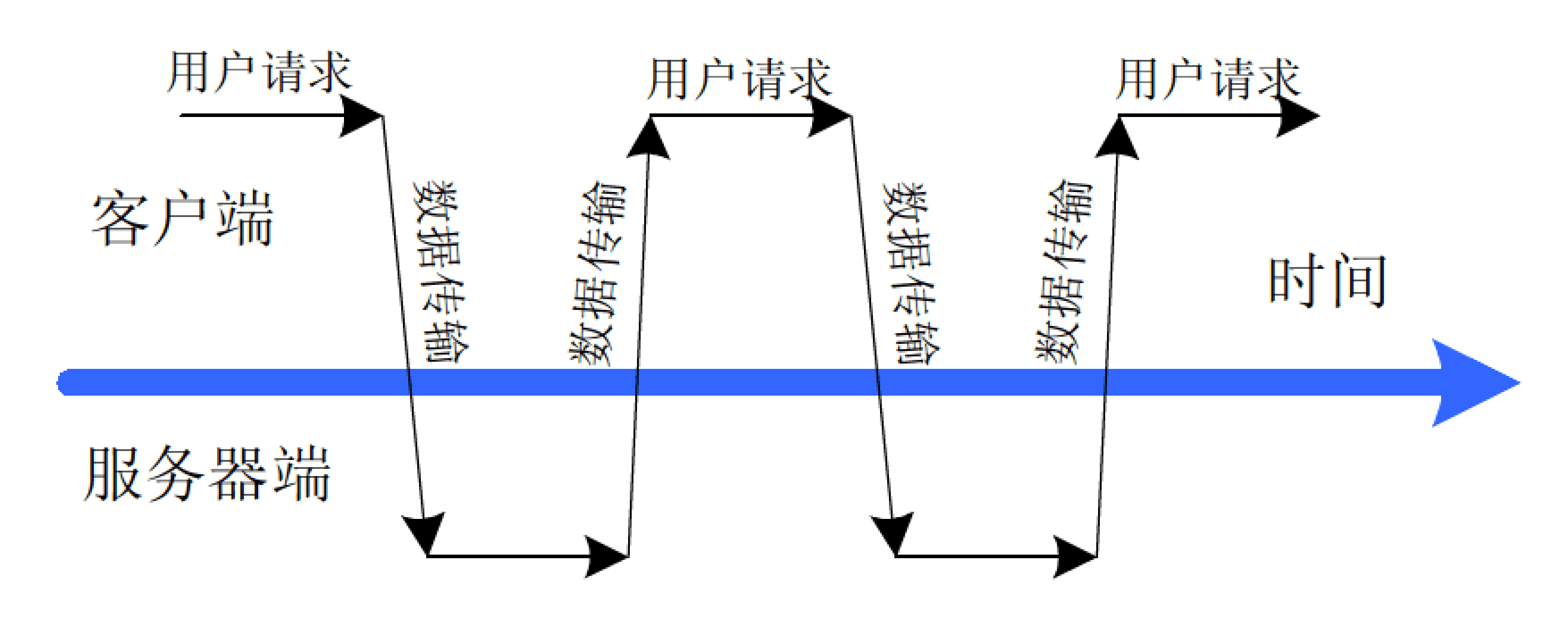


图 8-6　传统 Web 的同步交互模式

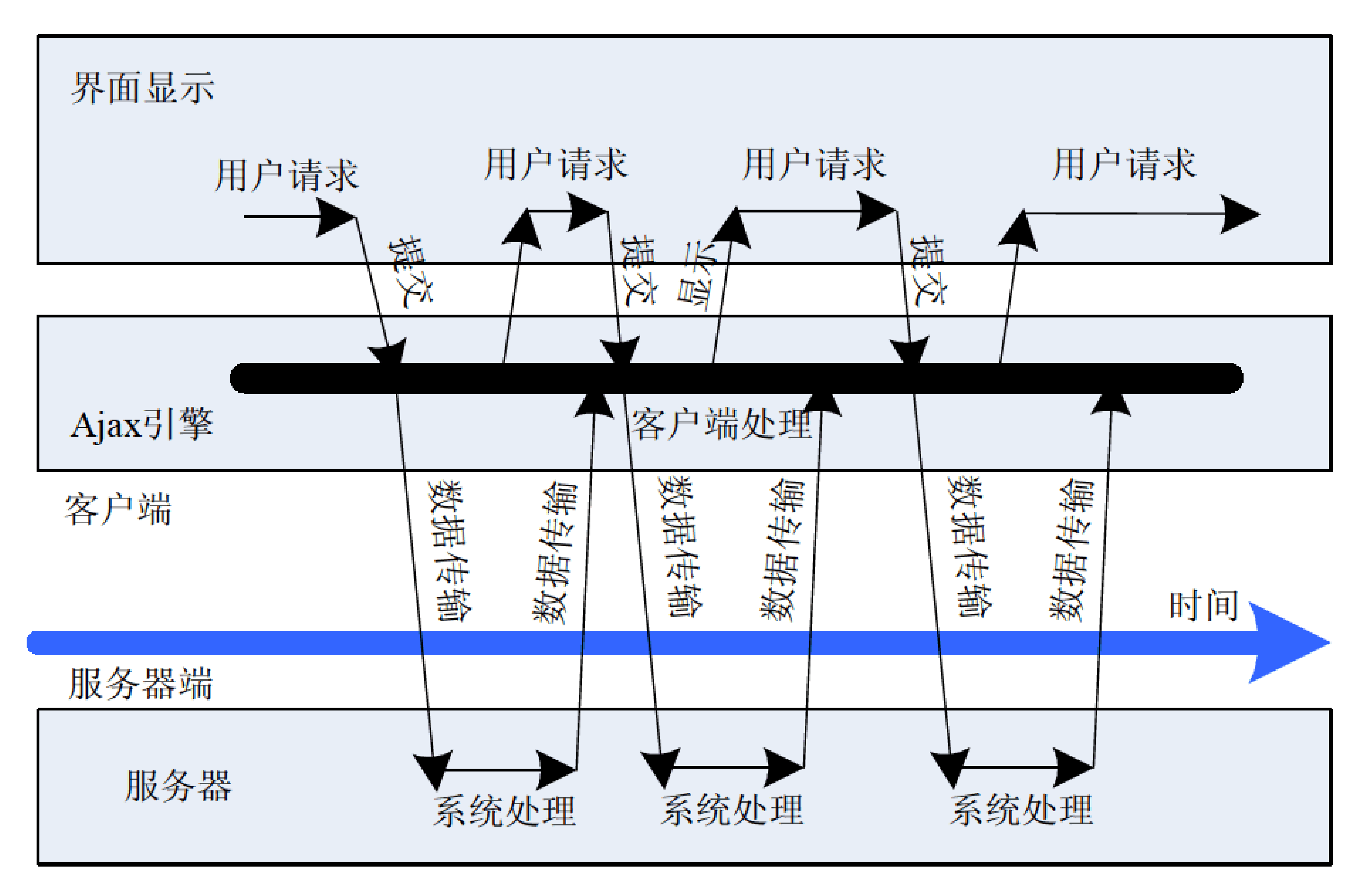


图 8-7　基于 Ajax 的 Web 异步交互模式

从图 8-6 及图 8-7 的分析和比较可以发现，使用 Ajax 的最大优点就是能在不刷新整个页面的前提下维护数据。这使得 Web 应用程序更为迅捷地响应用户交互，并避免了在网络上传输那些没有改变的信息。

Ajax 不需要任何浏览器插件，但需要用户允许 JavaScript 在浏览器上执行，因此 Ajax 应用程序必须在众多不同的浏览器和平台上经过严格的测试。随着 Ajax 的成熟，一些简化 Ajax 使用方法的程序库也相继问世。同样，也出现了另一种辅助程序设计的技术，为那些不支持 JavaScript 的用户提供替代功能。

对于应用 Ajax 而言，最主要的问题是：它可能破坏浏览器后退按钮的正常行为(参见 Jakob Nielsens 的 *1999 Top-10 New Mistakes of Web Design*)。在动态更新页面的情况下，用户无法回到前一个页面状态，因为浏览器仅能记忆历史记录中的静态页面。一个被完整读入的页面与一个已经被动态修改过的页面之间的差别非常微妙；用户通常会希望单击后退按钮能够取消他们的前一次操作，但是在 Ajax 应用程序中，这将无法实现。开发者们想出了种种办法来解决这个问题，大多数都是在用户单击后退按钮访问历史记录时，通过创建或使用一个隐藏的 IFRAME 来重现页面上的变更。例如，当用户在 Google 地图中单击后退按钮时，它在一个隐藏的 IFRAME 中进行搜索，然后将搜索结果反映到 Ajax 元素上，

以便将应用程序状态恢复到当时的状态。

进行 Ajax 开发时，网络延迟——用户发出请求到服务器发出响应之间的间隔——需要慎重考虑。不给予用户明确的回应，没有恰当的预读数据，或者对 XMLHttpRequest 的不恰当处理，都会使用户感到延迟，这是用户不希望看到的，也是他们无法理解的。对此，通常的解决方案是，使用一个可视化的组件来告诉用户系统正在进行后台操作并且正在读取数据和内容。

由此可见，Ajax 使得 Web 应用更加动态，带来了更高的智能，并且提供了表现能力丰富的 Ajax UI 组件。这样一类新型的 Web 应用叫作 RIA(Rich Internet Application)应用。除了 Ajax，还包括有 Flash 等技术。

与 20 世纪 90 年代末的 DHTML 相比，Ajax 更加强调符合真正的 Web 标准的开发方式。Ajax 对于现有基于 Web 标准技术的利用程度比 DHTML 高出了很多。而 DHTML 声名狼藉，最终失败的最大原因就在于其不重视基于真正的 Web 标准来做开发。

DHTML 其实是浏览器大战时代微软和 Netscape 为了吸引公众眼球而制造的一个名词，并没有得到 W3C 的认可。并且经常被开发人员滥用，制造出一大堆不符合真正的 Web 标准的 JavaScript 脚本和 HTML 标记，常常只能运行在某种特定的浏览器中(主要是 IE)。

DHTML 总是过于注重各种花哨的视觉效果，而 Ajax 最关注的问题则是真正改善 Web 应用可用性，这正是 Ajax 技术诞生的使命，甚至也正是 JavaScript 脚本语言诞生的使命。跨浏览器自然是 Web 应用可用性的重要组成部分，只有基于真正的 Web 标准来做开发，才有可能跨浏览器为用户提供一致的交互体验。而跨浏览器仅仅是基于真正的 Web 标准做开发的一个原因。另外一个原因是，唯有这样，才能充分地利用 Web 标准发展的成果(例如上述的 SVG、E4X 等符合标准的技术)，并且创建出向后兼容的 Web 应用。向后兼容的意思就是现在构建的 Web 应用，当未来用户都使用浏览器的新版本(如 IE 10.0)之后，不必再加以修改就能直接运行在这些新版本之上。这样可以降低 Web 应用的维护成本，并且可以真正达到改善可用性、使用户获得更好的交互体验的目标(想想看，假设用户将自己的浏览器升级为 IE 10.0，并且访问一个曾经去过的网站，突然发现网站的某个功能失效了，会有什么感觉？)。经验丰富的 Web 开发者都知道，以前专门为 IE 5.0 开发的 Web 应用，尤其是使用了很多 JavaScript 的应用，不加以修改和重新测试就运行在 IE 6.0 上几乎是不可能的。在这里就是没有做到向后兼容。而 Ajax 技术则会使得这些问题都不复存在。

8.3.4　Ajax 开发

Ajax 的核心是 JavaScript 对象 XmlHttpRequest。该对象首次在 IE 5 中引入，它支持异步请求。简而言之，XmlHttpRequest 可以利用 JavaScript 向服务器提出请求并处理响应，而不阻塞用户。在创建 Web 站点时，在客户端执行屏幕更新为用户提供了很大的灵活性。下面是使用 Ajax 在某个购物网站中能实现的功能：

- 动态更新购物车的物品总数，无须用户单击 Update 并等待服务器重新发送整个页面。
- 提升站点的性能，这是通过减少从服务器下载的数据量而实现的。例如，在某购

物车页面，当更新篮子中的一项物品的数量时，会重新载入整个页面，这必须下载整个页面的数据。但如果使用 Ajax 计算新的总量，服务器只会返回新的总量值，因此所需的带宽仅为原来的百分之一，这就消除了每次用户输入时的页面刷新。例如，在 Ajax 中，如果用户在分页列表上单击 Next 按钮，则服务器数据只刷新列表而不是整个页面。

- 直接编辑表格数据，而不是要求用户导航到新的页面来编辑数据。对于 Ajax，当用户单击 Edit 时，可以将静态表格刷新为内容可编辑的表格。用户单击 Done 按钮之后，就可以发出一个 Ajax 请求来更新服务器，并刷新表格，使其包含静态、只读的数据。

以上的功能很诱人，然而，开发和调试 Ajax 风格的 Web 应用程序是一项非常艰难的工作。要编写一个丰富的 Web UI，开发者需要详细地掌握 DHTML 和 JavaScript，并且还要掌握各种浏览器之间在设计细节上的不同。然而没有什么工具能够简化这些应用程序的设计和开发。最后，调试和测试这些应用程序会变得异常困难。微软致力于简化 Ajax 风格 Web 应用的开发，并提供丰富的、可交互的和个性化的用户体验。开发者可以对客户端脚本不甚了解，却可以很容易地开发和调试这种应用程序。出于这一目的，微软发布了 Atlas。它为开发者带来了如下特性：Atlas 客户端脚本框架、Atlas 的 ASP.NET 服务器控件、ASP.NET Web Services 集成、Atlas 的 ASP.NET 构建块和客户端构建块服务。

1. 客户端脚本框架

Atlas 客户端脚本框架是可扩展的，百分之百面向对象的 JavaScript 客户端脚本框架，允许开发者很容易地构建拥有丰富的 UI 工能并且可以连接 Web Services 的 Ajax 风格的浏览器应用程序。使用 Atlas，开发者可以使用 DHTML、JavaScript 和 XMLHTTP 来编写 Web 应用程序，而无须掌握这些技术的细节。

Atlas 客户端脚本框架可以在所有的现代浏览器上运行，而不需要 Web 服务器。它还完全不需要安装，只要在页面中引用正确的脚本文件即可。

Atlas 客户端脚本框架包含下列组件。

- 一个可扩展的新框架，其中为 JavaScript 添加了很多新特性，如生存期管理、集成、多播事件处理器和接口。
- 一个基础类库，提供了通用特性，如丰富的字符串操作功能、计时器和运行任务等。
- 一个 UI 框架，可以跨浏览器实现动态行为。
- 一个网络栈，用于简化对服务器的连接和对 Web Services 的访问。

2. ASP.NET 服务器控件

微软为 ASP.NET 应用程序专门设计了一组 Ajax 风格的服务器控件，并且加强了现有的 ASP.NET 页面框架和控件，以便支持 Atlas 客户端脚本框架。

ASP.NET 2.0 中有一项称作异步客户端回调的新特性，使得构建无中断的页面变得很容易。异步客户端回调包装了 XMLHTTP，能够在很多浏览器上工作。ASP.NET 本身包括

了很多使用回调的控件，包括具有客户端分页和排序功能的 GridView 和 DetalsView 控件，以及 TreeView 空间的虚拟列表支持。Atlas 客户端脚本框架将完全支持 ASP.NET 2.0 回调，但微软希望进一步增强浏览器和服务器之间的集成性。例如，可以将 Atlas 客户端控件的数据绑定，指定为服务器上的 ASP.NET 数据源控件，并且可以从客户端异步地控制 Web 页面的个性化特征。

3. ASP.NET Web Services 集成

和任何客户端应用程序一样，一个 Ajax 风格的 Web 应用程序通常也需要访问 Web 服务器的一些功能。Atlas 应用程序连接服务器的模型和其他平台类似，都是使用 Web Services 来实现。

通过 ASP.NET Web Services 集成，Atlas 应用程序将可以在任何支持 XMLHTTP 的浏览器上通过 Atlas 客户端用本框架来直接访问任何宿主于 ASP.NET 的 asmx 或 Indigo 服务。该框架将会自动处理和代理脚本到对象、对象到脚本的序列化问题。通过使用 Web Services 集成，开发者可以使用单一的编程模型来编写 Web Services，并且在任何应用程序中使用它们，不论是基于浏览器的站点上还是智能客户端应用程序中。

4. ASP.NET 构建块服务

在 ASP.NET 2.0 中，微软构建了一组丰富的构建块服务(Building Block Services)，这使得构建强大、个性化的 Web 应用程序变得不可思议的简单。这些构建块极大地降低了在开发通用的 Web 应用程序过程中需要编写的代码数量，比如管理用户、通过角色验证用户和存储用户的个性化设置信息等。

使用 Atlas，可以在任何浏览器上的任何客户端应用程序中像访问 Web Services 那样访问这些功能。例如，如果正在开发一个显示用户的 TO-DO 项目的站点，就可以使用 ASP.NET 的 Profile 服务来将它们存放在服务器上的用户自定义配置文件中。这样即使用户从一台计算机上转移到另一台计算机上，也同样可以访问这些项目。

微软将提供的服务包括(全部是基于 ASP.NET 2.0 的)以下几个方面。

- Profile：在服务器上存放每个用户特有的数据；
- UI 个性化：在服务器上存放个性化的 UI 设置信息；
- 验证：验证用户；
- 角色：基于用户的角色验证用户任务和提供不同的 UI。

由于这些构建块是服务器端的，开发者需要对它们应用和其他站点一样的安全模型。这些服务不需要客户端下的任何东西——只要在浏览器中引用脚本代理即可。

所有的 ASP.NET 2.0 构建块服务都是可插拔的，这使用一种通用的提供者模型可扩展模式在后台实现。微软提供的内建提供程序允许开发者使用 SQL Server 数据库或 Active Directory 作为存储容器，开发者也可以很容易地插接自己的提供程序。例如，希望使用集群而不是数据库服务器来存放用户的配置文件，这时只需提供程序插接即可。

5. 客户端构建块服务

除了 DHTML、JScript 和 XMLHTTP，微软还提供了一组附加的服务来加强客户端的功能并提供增强的体验。

对于这样的服务，本地浏览器缓存就是一个很好的例子。当启用了本地浏览器缓存时，Web 站点就可以将内容存储到缓存中，并在需要的时候很快地取出。但浏览器并未提供向缓存中存放数据的 API，而且像 Google Map 或 OWA 这样的应用程序不得不通过很多工作产生一个唯一的 URL 才能使浏览器缓存它。在 Atlas 中，微软提供了可编程的本地存储/缓存，因此应用程序可以很方便、有效并且安全地在本地缓存数据。

【实例 8-4】一个简单的 Ajax 实例

为了实际考察 Ajax 的实际效果，这里给出一个简单的 Ajax 的实例，其建立步骤如下。

(1) 首先在 VS 2005(或 VS.NET 2003)建立一个普通的 Web 项目，或者未安装 VS 开发工具，直接用记事本或 EditPlus 建立以下文件也可以。

(2) 在项目中加入一个客户端页面 ex8_4.html，并将此页面设定为起始页，这个页面会向 Web 服务器发出非同步呼叫请求，并且将服务器回传的信息更新到网页中，其代码如下。

ex8_4.html

```
DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
  <HEAD>
    <SCRIPT language="javascript">
      var XmlHttp=new ActiveXObject("Microsoft.XMLhttp");
      function sendAJAX()
      { XmlHttp.Open("POST","ex8_4.aspx",true);
        XmlHttp.send(null);
        XmlHttp.onreadystatechange=ServerProcess;
      }
      function ServerProcess()
      { if (XmlHttp.readystate==4 || XmlHttp.readystate=='complete')
          document.getElementById('nameList').innerHTML=XmlHttp.responsetext;
      }
      setInterval('sendAJAX()',1000);
    <SCRIPT>
  </HEAD>
  <BODY>
    <div id="nameList"></div>
  </BODY>
</HTML>
```

(3) 在项目中加入一个 ex8_4.aspx.cs 网页，其中 ex8_4.aspx.cs 不需要添加任何代码。ex8_4.aspx.cs 文件如下。

ex8_4.aspx.cs

```
using System;
```

```
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using System.Diagnostics;
public partial class Server : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        PerformanceCounter myMemory = new PerformanceCounter();
        myMemory.CategoryName = "Memory";
        myMemory.CounterName = "Available KBytes";
        string txtResult = "-->服务器可以用内存大小：" +
        myMemory.NextValue().ToString() + "kb";
        Response.Write(DateTime.Now.ToLongTimeString() + txtResult);
    }
}
```

本实例运行后的浏览器中的数据每隔一秒钟变化一次，但整个页面却不刷新，其显示效果如图 8-8 所示。

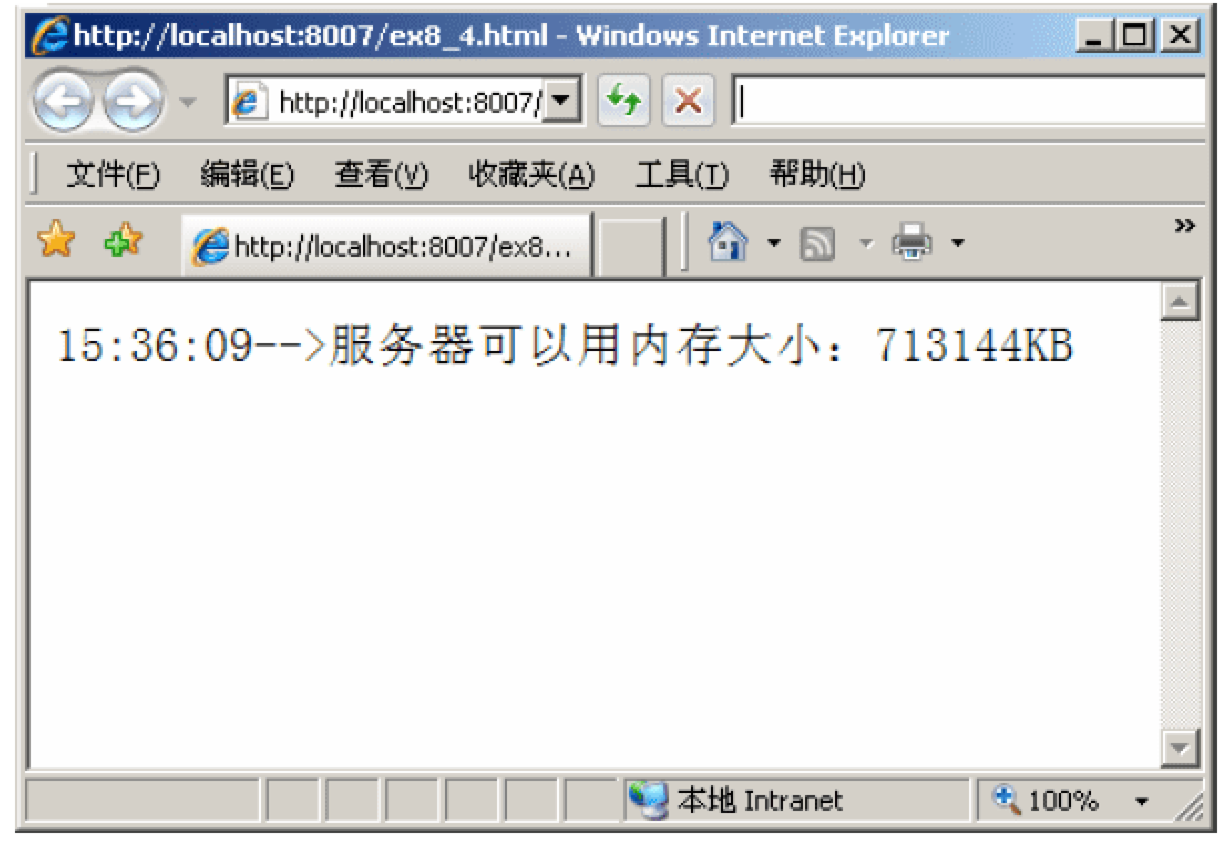


图 8-8　一个简单的 Ajax 实例

注释：

限于篇幅，此处不深入介绍具体的开发方法，如果希望深入学习有关 Ajax 的开发，可参考专门的书籍。

一切皆有可能！但愿 Ajax 能够激发创意，让读者尝试开发属于自己的基于 Ajax 的网站。

8.4　功能强大的客户端开发框架

JavaScript 目前被广泛地应用于 Web 开发中，随着 HTML5 技术的发展，JavaScript 在未来还将有更大的发展和应用空间。行业分析机构 RedMonk 近期的一份调查显示，JavaScript 目前在最受欢迎编程语言排行榜中排名第一。

而实际开发过程中，往往会考虑借助开发框架，这样可以快速建立应用，其中 jQuery 是最流行的 JavaScript 库。据调查，互联网中近一半的网站都使用了 jQuery。使用 jQuery，开发者的编码工作将大大减少，而大量的 jQuery 插件，也使得开发者可以轻易实现很多绚丽的效果。但是在 Web 开发中，并不是用到 JavaScript 的地方都适合使用 jQuery。以下对

常用的开发框架进行介绍。

8.4.1　jQuery 框架

1. jQuery 的优点

(1) jQuery 实现脚本与页面的分离

在 HTML 代码中，我们还经常看到类似这样的代码：<"form id="myform" onsubmit=return validate();">。即使 validate()函数可以被放置在一个外部文件中，实际上我们依然是把页面与逻辑和事件混杂在一起。jQuery 可以将这两部分分离。借助于 jQuery，页面代码为：<form id="myform">。接下来，一个单独的 JS 文件将包含以下事件提交代码：

```
$("myform").submit(function() {
...your code here
)}
```

这样可以实现灵活性非常强的清晰页面代码。jQuery 让 JavaScript 代码从 HTML 页面代码中分离出来，就像 CSS 能让样式代码与页面代码分开一样。

(2) 最少的代码做最多的事情

最少的代码做最多的事情，这是 jQuery 的口号，而且名副其实。使用它的高级 selector，开发者只需编写几行代码就能实现令人惊奇的效果。开发者无须过于担忧浏览器差异，此外它还完全支持 Ajax，而且拥有许多提高开发者编程效率的其他抽象概念。jQuery 把 JavaScript 带到了一个更高的层次。

使用 JavaScript 实现获取元素的值的代码为：

```
document.getElementById('elementid').value
```

而实现相同功能的 jQuery 代码则为：

```
$('#elementid').val();
```

以下是另一个简单的示例：

```
$("p.neat").addClass("ohmy").show("slow");
```

通过以上简短的代码，开发者可以遍历“neat”类中所有的<p>元素，然后向其增加“ohmy”类，同时以动画效果缓缓显示每一个段落。开发者无须检查客户端浏览器类型，无须编写循环代码，无需编写复杂的动画函数，仅仅通过一行代码就能实现上述效果。

(3) 性能

在所有 JavaScript 框架中，jQuery 的性能表现优异。尽管不同版本拥有众多新功能，其最精简版本只有 18kb 大小。jQuery 的每一个版本都有重大性能提高。如果将其与新一代具有更快 JavaScript 引擎的浏览器(如火狐和谷歌 Chrome 等)配合使用，在创建富体验 Web 应用时将拥有全新速度优势。

(4) 它是一个“标准”

之所以使用引号，是因为 jQuery 并非一个官方标准。但谷歌不但自己使用它，还提供给用户使用。另外戴尔、新闻聚合网站 Digg、WordPress、Mozilla 和许多其他厂商也在使用它。微软甚至将它整合到 Visual Studio 中，如此多的重量级厂商都共同支持了该框架。

(5) 插件

基于 jQuery 开发的插件目前已有数千个。开发者可使用插件来进行表单确认、图表种类、字段提示、动画、进度条等任务。而且，jQuery 正在主动与竞争对手合作，如 Prototype。它们似乎在推进 JavaScript 的整体发展，而不仅仅是在图谋一己之私。

(6) 节省开发者学习时间

当然要想真正学习 jQuery，开发者还是需要投入一点时间，尤其是如果编写大量代码或自主插件的话，更是如此。但是，开发者可以采取“各个击破”的方式，而且 jQuery 提供了大量示例代码，入门是一件非常容易的事情。笔者建议开发者在自己编写某类代码前，首先看一下是否有类似插件，然后看一下实际的插件代码，了解一下其工作原理。简而言之，学习 jQuery 不需要开发者投入太多，就能够迅速开始开发工作，然后逐渐提高技巧。

(7) 让 JavaScript 编程变得有趣

使用 jQuery 是一件充满乐趣的事情。它简洁而强大，开发者能够迅速得到自己想要的结果，同时也解决了许多 JavaScript 问题和难题。通过一些基础性的改进，开发者可以真正去思考开发下一代 Web 应用，不再因为语言或工具的差劲而烦恼。“最少的代码做最多的事情”的口号名副其实。

2. jQuery 的缺点

(1) 不能向后兼容

新版本不能兼容早期的版本。比如有些新版本不再支持一些 selector，新版本只是简单地将这些功能做了移除，这可能会影响到开发者已经编写好的代码或插件。

(2) 插件兼容性

当新版 jQuery 推出后，如果开发者想升级的话，要看插件是否被支持。通常情况下，新版本中现有插件可能无法正常使用。开发者使用的插件越多，这种情况发生的概率也越高。

(3) 插件间的冲突现象

在同一页面上使用多个插件时，很容易碰到冲突现象，尤其是这些插件依赖相同事件或 selector 时最为明显。这虽然不是 jQuery 自身的问题，但却是一个难于调试和解决的问题。

(4) jQuery 的稳定性

此处指的是其版本发布策略。比如 jQuery 1.3 版发布后仅过数天，就发布了一个漏洞修正版 jQuery1.3.1。其中就移除了对某些功能的支持，这可能会影响许多现有代码的正常运行。

(5) 对动画和特效的支持度

在大型框架中，jQuery 核心代码库对动画和特效的支持相对较差，但是实际上这不是

一个问题。目前在这方面有一个单独的 jQuery UI 项目和众多插件来弥补此点。

对于是否要学习某个 JavaScript 框架，并困惑于选择哪一个框架，那么可以首先选择 jQuery，因为它是最稳妥和最具回报性的选择。

8.4.2　ExtJs

ExtJs 可以用来开发 RIA 也即富客户端的 Ajax 应用，这是一个用 JavaScript 写的、主要用于创建前端用户界面、与后台技术无关的前端 Ajax 框架。因此，可以把 ExtJs 用在.Net、Java、Php 等各种开发语言开发的应用中。ExtJs 最开始基于 YUI 技术，由开发人员 JackSlocum 开发，通过参考 JavaSwing 等机制来组织可视化组件，无论从 UI 界面上 CSS 样式的应用，还是数据解析上的异常处理，其都可算是一款不可多得的 JavaScript 客户端技术的精品。相对来说，ExtJs 要比开发者直接针对 DOM、W3C 对象模型开发 UI 组件轻松。

ExtJs 的功能丰富，无论是界面之美，还是功能之强，它都高居榜首。其表格控件所提供的编辑功能强大，主要包括单选行、多选行、高亮显示选中的行、拖曳改变列宽度、按列排序、自动生成行号、支持 checkbox 全选、动态选择显示哪些列、支持本地以及远程分页，可以对单元格按照自己的想法进行渲染、添加新行、删除一或多行、提示多行数据、拖曳改变表格大小、表格之间拖曳一行或多行，甚至可以在树和表格之间进行拖曳。

JQuery、Prototype 和 YUI 都属于非常核心的 JS 库。虽然 YUI 及最近的 JQuery，都给自己构建了一系列的 UI 器件(Widget)，不过却没有一个真正的整合好的和完整的程序开发平台。哪怕是这些底层的核心库已经非常不错了，但当投入到真正的开发环境中，依然需要开发者做大量的工作去完善很多缺失之处。而 ExtJs 就是要填补这些缺口。主流开源框架中只有 Dojo 像 ExtJs 一样，尝试着提供整合的开发平台。相比 Dojo 这个出色的工具包，我们认为 ExtJs 能提供一个黏合度更高的应用程序框架。ExtJs 的各个组件在设计之时就要求和其他 ExtJs 组件组合在一起工作是无缝合作的。这种流畅的互通性，离不开一个紧密合作的团队，还必须时刻强调设计和开发这两方面目标上的统一，而这点是很多开源项目未能做到的。

8.4.3　Flex

Flex 是一个高效、免费的开源框架，可用于构建具有表现力的 Web 应用程序，这些应用程序利用 Adobe Flash Player 和 Adobe AIR，可运行时跨浏览器、桌面和操作系统实现一致的部署。使用 Flex 创建的 RIA 可运行于安装了 Adobe Flash Player 软件的浏览器中，或在浏览器外运行于跨操作系统的 Adobe AIR 上，它们可以跨所有主要浏览器，在桌面上实现一致的运行。连接到 Internet 的计算机中超过 98%的装有 Flash Player，这是一个企业级客户端运行时，它的高级矢量图形能处理要求最高、数据密集型应用程序，同时达到桌面应用程序的执行速度。通过利用 AIR，Flex 应用程序可以访问本地数据和系统资源。

运用 Flash 是完全可以做到 Flex 的效果的，Flex 的存在有何意义？首先，为了迎合更多的开发者，Flash 最初是为了动画设计者而设计的，其界面与程序开发人员的使用习惯大不相同，为了吸引更多的程序员而推出了 Flex，它用非常简单的 MXML 来描述界面，提

供 Jsp/Asp/PHP 程序人员与编辑 HTML 相似的操作界面，且 MXML 更加规范化、标准化。另外，微软推出了新的语言 XAML，它是一种界面描述语言，与之相应的就是 Smart Client 和与 Flex 非常相似的 SilverLight。MXML 和 XAML 也很相似。这是人机交互技术进步的重要体现，即内部逻辑与外部界面交互相分离。当编译 Flash 程序时，Flash 开发环境把所有的可视化元素、时间轴指令和 ActionScript 中的业务逻辑编译为 SWF 文件。同样的，Flex 程序中的 MXML 和 ActionScript 代码首先全部被转换为 ActionScript，然后编译为 SWF 文件，当此 SWF 文件被部署到服务器上时，使用者可以从服务器上获取到这个程序。

Flex 的设计目标是让程序员更快更简单地开发 RIA 应用。尽管用 Flex 开发 RIA 有多种形式，但现在主流的架构是：Flex 作为客户端开发技术，Java、PHP、Asp、Ruby 等技术作为服务器端开发语言。在多层式开发模型中，Flex 应用属于表现层，采用 GUI 界面开发。它具有多种组件，可实现 Web Services、远程对象、拖放、列排序、图表等功能；且内建动画效果和其他简单互动界面等。相对于基于 HTML 的应用(如 PHP、ASP、JSP、ColdFusion 及 CFMX 等)在执行每个请求时都需要执行服务器端的交互；Flex 的客户端只需要载入一次，因此其工作流被大大改善。Flex 的语言和文件结构也试图把应用程序的逻辑从设计中分离出来。Flex 服务器也是客户端和 XML Web Services 及远程对象(Coldfusion CFCs，或 Java 类，等支持 Action Message Format 的其他对象)之间通信的通路。一般认为与 Flex 平行的是 OpenLaszlo 和 AJAX 技术。

作为新一代的富客户端互联网技术的佼佼者，Flex 这种技术已经被越来越多的公司所采用，被越来越多的用户和程序员所接受。其优势在于：可以让普通程序员开发制作 Flash、界面表现能力强、构架 RIA 富客户端应用时，实现异步调用、界面无刷新、浏览器兼容性等多项难题。支持流媒体，可实现跨平台，对底层的可操作性，如可以调用摄像头实现视频等。具有 Flex 官方样式配置工具，可以在线配置 Flex 应用程序各种控件的外观样式，可用任何 Web 编程平台作为后台数据访问层，如.NET、PHP、Jsp、WebService 等。

8.4.4　其他框架

除了上面重点介绍的框架外，还包括以下一些其他类似框架。

(1) Yii：它是一个高性能的、开发 Web 2.0 应用程序最好的 PHP 框架。

(2) 52 Framework：它支持 HTML5 和 CSS 3，支持目前所有的浏览器。该框架充分利用了 HTML5 所有的优势。在网页设计师的世界中，CSS 3 是非常酷的东西，使用 CSS 3 可以节省网页设计和布局的时间。在开发中可以使用 CSS 3 所有的特性，如文本框阴影、圆角和动画等。

(3) YAML：全称为 Yet Another Multicolumn Layout，它是一个用于创建现代、灵活的浮动层的 HTML(XHTML)/CSS 框架。

(4) Zoop：自从 2001 年发布以来，已在大量不同环境中使用。它基于坚实的 MVC 原理，包括表现层、逻辑层和数据层。它具有高效、模块化和可扩展的特性，在轻量级和功能强大之间达到了惊人的平衡。

(5) Symfony：是一个开源的 PHP Web 应用程序开发框架。最初由 Sensio 实验室用来

为自己的客户开发网站，于 2005 年被该机构基于 MIT 许可开源。现在它已经成为 PHP 开发中领先的框架之一。

(6) CakePHP：一个 PHP 快速开发框架，提供了一个用于开发、维护和部署应用程序的可扩展结构。在配置范例中采用了应用普遍的设计模式，如 MVC 和 ORM 等，可以降低开发成本，帮助开发者减少代码的编写量。

(7) MooTools：一个简洁、模块化、面向对象的 JavaScript 框架。它能够更快更简单地编写可扩展和兼容性强的 JavaScript 代码。MooTools 与 ProtoTypeJS 相类似，语法几乎一样。但它提供的功能要比 ProtoTypeJS 多，而且更强大，如增加了动画特效、拖放操作等。

(8) Web2py：一个免费、开源的全栈框架，可用于快速开发可扩展、安全、轻便、数据库驱动、基于 Web 的应用程序。该框架是用 Python 编写的。

8.5　移动开发

8.5.1　移动开发简介

20 世纪 90 年代末，随着移动互联网的发展，国外一些媒体给出了一个“第五次科技革命”的概念。而随着 iPhone 和 Android 等智能手机的日渐流行和 iPad 等平板电脑的出现，移动互联网的潜力和趋势也愈发显现，“第五次科技革命”——移动互联网正式走进了开发者的视线。而针对移动互联网原动力——移动设备的 Web 开发越来越受到关注，国内外很多公司也开始重视面向所有移动设备的 Web 开发。

1. 移动开发面临的挑战

(1) 相对封闭性

研究发现，在移动互联网领域存在着很多由大企业牵头构建的相对封闭生态圈。例如，诺基亚、苹果等终端制造企业构建了包括从终端生产、操作系统提供到最后的应用商店(如诺基亚的 OVI、苹果的 APP Store)、内容服务在内的相对封闭生态圈。又例如以手机操作系统为核心，也形成了一系列的封闭生态圈，例如 WM、塞班、Android、Palm 等，各操作系统的通信录、应用等相互之间不兼容，这就构建了一定的壁垒，使得用户稳定在一个圈子当中使用。

移动互联网的这些封闭生态圈的重要特点是各个生态圈之间相互实力较为接近，客户群规模也比较接近。而反观传统互联网，则是相对开放的，终端基本基于 X86 架构，操作系统基本是由微软的 Windows 主导，由于底层技术较为统一，因此在业务模式上能形成开放的状态。

移动互联网正因为有封闭性存在，一系列盈利模式因此应运而生，如苹果的 APP Store 等就是借助于其通过终端和操作系统所圈定的用户而实现盈利的。但对于一些独立第三方公司而言，一个个实力相近的封闭生态圈是一场灾难，如在应用开发领域，比较传统互联

网，他们需要开发出更多版本应用程序，甚至有的时候，在排他性协议面前，他们需要抉择在哪个生态圈当中发展。

(2) 个人性

根据一项调查的结论显示，92.4%连接到移动互联网的终端(主要是手机)基本上是供一个人使用的。换句话说，即当移动互联网服务提供商发现某一特定的终端连接时(可以通过手机号、IEMI 号等进行判断)，可以直接判断出是谁正在上网，这体现了移动互联网的一种个人性特征。而传统互联网的终端共享性较高，网吧、家庭共享电脑等，使得运营商无法通过终端直接追溯到是谁在使用、使用者的特征如何。

移动互联网个人性的特征，在具体应用过程中，主要体现在业务和服务层面。用户在使用移动互联网服务时，服务器端检测用户的手机号，将此作为用户的账号予以登录，直接进行业务的提供，并且也可以分析用户既往的消费特征、用户的位置信息等，从而向用户提供更为精确的个性化服务。

(3) 终端类型众多

操作系统、终端显示分辨率、键盘类型和处理速度是极其丰富的，例如依据输入方式的维度可分为触屏、普通尺寸和全尺寸键盘；依据手机操作系统可分为 WM、SmartPhone、S60、Android、Palm、BlackBerry、iOS 等；依据屏幕分辨率维度可分为 220×176、320×240、400×240、640×480 等；依据处理器的速度，从 200MHZ 到 1GMHZ 不等。而传统互联网领域、主流 PC 终端的操作系统基本类似，分辨率基本类似，处理器的效能基本类似，所形成的差异主要在界面以及所蕴含的一些其他功能上，但是基本形态(操作系统、分辨率以及处理器速度)则基本相似。

众多的移动终端给移动互联网服务提供商带来了挑战。例如做一个页面，在传统互联网模式下，一般需要考虑宽屏和普屏两种类型的展示，而在移动互联网下，需要考虑较多类型的分辨率的屏幕适配问题，考虑不同处理器的手机能否实现该页面展示的问题。而对于不同的应用程序开发而言，则更是如此，需要针对各款终端的分辨率、处理器的处理能力、内存等进行相应的调整。

(4) 入口的重要性

可以看到，无论移动终端是怎样的输入方式(如触屏、普通键盘和全尺寸键盘)，其比较 PC 终端的大尺寸键盘而言，输入仍然较为烦琐。根据一项 2009 年的调查，55.1%的传统互联网使用者经常性地在互联网中输入网址，这类用户觉得直接输入一些常用网址，比通过收藏夹更为便利；但是同样的调研显示，仅有 13.1%的移动互联网用户经常性地在其手机上输入网址，而更多的客户则是使用自己收藏夹中的网址、UC 中所推荐的网址，或者从一些客户端软件直接进入等。

在这样的情况下，抢占客户入口，使得客户通过这些入口能够直接访问移动互联网就显得十分重要了。这些入口包括客户端程序、垂直性网站等。目前，一些客户端已经成为入口的平台，例如 UC 已成为诸多网站流量形成的依赖。据了解，某网站停止使用 UC 作为入口接入之后，其访问量当天下降了 64%，由此可见，这些平台性客户端的重要性，甚至可以说他们“劫持”了国内相当部分的网站。

(5) 流量限制内容

目前国内手机上网的计费方式以流量计费为主，这就使得用户使用移动互联网的行为同流量相结合。根据一项调研显示，76.1%的手机上网客户对流量是较为关注的，办理了套餐，并且表示不愿意每月所使用的流量超出套餐费，这也就是运营商内部所经常提及的“20 日效应”产生的主要原因，即每月 20 日之后，使用的数据流量迅速下降。同时可以看到以节约流量为卖点的第三方手机浏览器，如 UC 的迅猛发展，同其宣传口号“节约 99%的流量”密不可分。

在这样的情况下，丰富的多媒体内容难以展示，流量较小的文字或者图片成为适合展示的元素；同时，基于当前移动终端处理器的进步，可以在一些联网应用当中发展强客户端，将大多数处理放在本地执行，从而减少流量传输。

(6) 碎片化的应用场景

移动互联网尽管有丰富的应用，但其终端及展示界面的特性，决定了客户应用移动互联网的场景主要是一些碎片化的时间；而整段的时间，往往在应用展示较为丰富的传统互联网之上。这就要求移动互联网服务提供商构建能够满足用户在碎片化时间需求的应用，这些需求以及参考应用包括实时性较强，如股票；应急性需求，如导航；无聊性需求，如阅读新闻等；无缝性需求，即需要延续在传统互联网上的行为，不至于中断，如移动办公、网络游戏等。

2. 移动开发及其技术

移动 Web 开发的优点如下。

- 易于开发，新用户易上手，开发周期相对较短。
- 自动更新，只要服务器端更新后，所有移动设备也一起更新。
- 可充分利用现有 Web 内容。

一般来说，对于移动 Web 可以采取两种方式：专门开发一个独立的移动版本，或者使用 Media Type 和 Media Query 控制 Web 在移动浏览器的表现。相关的开发技术包括 WML、cHTML、XHTML Basic、XHTML MP 和 HTML5 等。

WML 是一种基于 XML 的语言，它是用于 WAP 网站的标记语言。它将网站分割为两部分：普通页面使用(X)HTML，而移动 Web 使用 WML。网站开发者想要做一个移动 Web 也不得不学习一种新的语言而不是转换技术，“一站式”的信条也被打破，用户不能访问其喜欢的网站且不得不发现这个网站的 WAP 版本——如果它们存在的话。

日本的 NTT 创建了一种称为 cHTML 的语言(compact HTML)，但是它并不能与 XHTML 和 WML 兼容。

由于这些与理想中的方案存在差距，W3C 创建了 XHTML Basic 1.0。它是 XHTML 1.1 的子集。由于 XHTML 1.1 将 XHTML 改善为小型的模块，一个子集就可以只包含一些必需的或者可以在低端移动设备上控制的基本的模块、元素和属性。XHTML Basic 为针对移动 Web 的标记语言提供基础的模块。与其基础的 XML 一样，它也被设计用于扩展。这正是结合了 WAP 和 NTT 的合并之后(也就是 OMA)的做法，创建了 cHTML 和 WML 的继承

者 XHTML Mobile Profile——它在 XHTML Basic 的基础上添加了一些在它们之前的版本中有的特性。XHTML Basic 和 XHTML MP 共存的状况看起来有些混乱，但是之后不久 W3C 就发布了 XHTML 1.1 版本，吸收了在 XHTML MP 中加入的一些特性。所以现在看来这两个版本差不多是一样的。

XHTML MP 是对 XHTML Basic 的一个扩展，所以 XHTML MP 有更好的适用性。而 XHTML MP 对于 Basic 最大的优势就是支持外部样式文件——虽然这会导致多一个 HTTP 请求。

对于最新的 HTML5，Mobile Webkit 是目前对标准支持最好的移动浏览器，它支持所有的 XHTML 特性，同时对 HTML5 的支持也非常棒。如果只针对 iPhone 和(或) Android，完全可以使用 HTML 5 来编码。事实上作为又一个很强劲的趋势，HTML 5 众望所归要成为下一代的网页标准，Google、Apple、Opera 和微软等互联网巨头一直在努力推广和推进 HTML 5。

正如之前所说的，XHTML Basic 支持了大部分在 XHTML 中定义的基础特性，所以对于大部分前端开发人员来说，开发一个基于 XHTML Basic 1.1 或 XHTML MP 的网站并不困难。但是由于移动设备厂商和设备都非常多，所以各个设备在对某个细节上可能会有差异。

W3C 存在的最大价值，是提供成熟而统一的解决方案，虽然 XHTML MP 成了事实上的标准，但是显然 XHTML Basic 功不可没，如果说两者并存尚容易让人混淆的话，希望在不久的将来，HTML 5 能够成为移动互联网中事实上的标准，这无疑将大大减少人们的困惑。但是由于现实中很难将所有的设备统一，这就造成实现方式必然存在差异。可以预见，XHTML Basic /MP 和 HTML 5 将成为两种并行的规范存在，将来也许不得不用 XHTML Basic/MP 为低端设备开发基础页面，同时使用 HTML 5 为 iPhone 和 Android 等系统实现富界面。

8.5.2　移动开发框架

目前，各种移动 Web 开发的框架也纷纷到来，主要包括以下几种。

- jQuery Mobile：是 jQuery 针对移动设备的版本。主要包括针对移动设备的 jQuery Core 和 jQuery UI。支持目前主流的移动操作系统(Android、iPhone、Symbian、Blackbery、webOS 等)。
- iUI：是一个 JavaScript 和 CSS 库，用于在网页中模拟 iPhone 的外观和感觉。在 Android 上 90%以上的功能是可以使用的，因为它们都是基于 WebKit 浏览器的系统。
- jQTouch：是一个用于移动 Web 开发的 jQuery 插件，支持 iPhone、iPod Touch 和其他一些基于 WebKit 的系统。
- Sencha Touch：可以让 Web App 看起来像 Native App。美丽的用户界面组件和丰富的数据管理，全部基于 HTML5 和 CSS 3 标准，全面兼容 Android 和 Apple iOS 设备。它是 ExtJs 整合 jqTouch 和 Raphaël 库的产物。

8.6　本章小结

本章首先介绍了 Web 的进化路径，为未来的发展指明了发展的方向，然后介绍了 Web 方面的新技术——XML、Ajax、开发框架技术和移动开发等。XML 这种元标记语言可以将数据和显示格式分开，为将来面向语义的 Web 奠定基础；Ajax 基于 XML，通过异步响应的工作方式为网页使用者带来了新的体验；开发框架为开发者提供了高效和快捷的实现手段；移动开发则揭示了未来 Web 开发的新方向。它们都为 Web 应用带来了很多新的特征，也为 Web 提供了更大的发展和想象空间。

读者可以通过本章的阅读了解这些技术的由来、基本特征和用途，为进一步的深入学习奠定基础。

8.7　思考和练习

1. 请归纳和总结 XML 与 HTML 的差异。HTML 会被 XML 替代吗？
2. 请问 XML 如何做才能像 HTML 一样显示出漂亮且激动人心的网页？
3. Ajax 的优越性体现在哪些方面？
4. 客户端开发框架对开发者意味着什么？
5. 移动开发与传统开发有什么差异？

参 考 文 献

[1] 郝兴伟．Web 技术导论．北京：清华大学出版社，2005.

[2] Rosenfeld, Louis, and Morville. *Information Architecture for the World Wide Web: Designing Large-Scale Web Sites*, O'Reilly, 2006.

[3] R.Fielding, J.Gettys, J.C.Mogul, H.Frystyk, L.Masinter, P.Leach, T.Berners-lee. *Hypertext Transfer Protocol - HTTP/1.1*, RFC2616, June 1999.

[4] Dave Raggett, et al. *HTML 4.01 Specification*, W3C Recommendation, December 1999

[5] Marcotte, Ethan. *Responsive Web Design*, A Book Apart, 2011.

[6] 王继成，武港山．WEB 应用开发原理与技术．北京：机械工业出版社，2003.

[7] 工作过程导向新理念丛书编委会. 网站服务器搭建与管理. 北京：清华大学出版社，2009.

[8] Gustafson, Aaron. *Adaptive Web Design,* East Readers, 2011.

[9] Pickering, Heydon. *Apps For All: Coding Accessible Web Applications*, Smashing Magazine GmbH, 2014.

[10] 严晨. 多媒体界面设计. 北京：电子工业出版社，2011.

[11] Allsopp , John. *Developing with Web Standards,* New Riders, 2010.

[12] Holzschlag, Molly. *Color for Websites*, RotoVision, 2001.

[13] Sauro and Lewis. *Quantifying the User Experiences*, Morgan Kaufmann, 2012.

[14] Goto, Kelly and Cotler, Emily. *Web ReDesign 2.0: Workflow that Works*, New Riders, 2004.

[15] [美]吉伦瓦特(Gillenwater，Z.M.). CSS 3 实用指南. 北京：人民邮电出版社，2012.

[16] Freeman, Elisabeth and Freeman, Eric. *Head First HTML with CSS & XHTML*, O'Reilly & Associates, 2005.

[17] 刘玉萍，刘增杰编著. 精通 HTML5 网页设计.北京：清华大学出版社，2013.

[18] Cederholm, Dan. *CSS3 For Web Designers,* A Book Apart, 2010.

[19] Powers, Shelley. *JavaScript Cookbook*, O'Reilly & Associates, 2010.

[20] Rauschmayer, Axel. *Speaking JavaScript, An In-Depth Guide for Programmers*, O'Reilly & Associates, 2014.

[21] Hassell, Jonathan. *Including your missing 20% by embedding web and mobile accessibility*, BSI British Standards Institution, 2014.

[22] 李东博. HTML5+CSS 3 从入门到精通. 北京：清华大学出版社，2014.

[23] 唐俊开. HTML5 移动 Web 开发指南. 北京：电子工业出版社，2012.

[24] 高怡新．XML 基础教程．北京：人民邮电出版社，2006.

[25] 李天平编著.项目中的.NET. 北京：电子工业出版社，2012.

[26] 埃克尔. Java 编程思想. 4 版 . 北京：机械工业出版社，2007.

[27] [美]奥特罗，劳伦斯，施宏斌译. jQuery 高级编程. 北京：清华大学出版社，2013.